DEVELOPMENT OF A KNOWLEDGE MODEL

FOR THE COMPUTER-AIDED DESIGN FOR RELIABILITY

OF ELECTRONIC PACKAGING SYSTEMS

A Dissertation
Presented to
The Academic Faculty

by

Injoong Kim

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Mechanical Engineering

Georgia Institute of Technology
April 2008

DEVELOPMENT OF A KNOWLEDGE MODEL

FOR THE COMPUTER-AIDED DESIGN FOR RELIABILITY

OF ELECTRONIC PACKAGING SYSTEMS

Approved by:

Dr. Suresh K. Sitaraman, Co-Advisor
School of Mechanical Engineering
Georgia Institute of Technology

Dr. Raghuram V. Pucha
School of Mechanical Engineering
Georgia Institute of Technology

Dr. Russell S. Peak, Co-Advisor
Product & Systems Lifecycle
Managemet Center
Georgia Institute of Technology

Dr. C. P. Wong
School of Materials Science & Engineering
Georgia Institute of Technology

Dr. Christiaan Paredis
School of Mechanical Engineering
Georgia Institute of Technology

Date Approved: December 10, 2007

To Kyunghwa, Jungmin, and Joomin

ACKNOWLEDGEMENTS

My achievement would not have been realized without the efforts of those who have given me precious help.

I give my sincere appreciation to my advisor, Dr. Suresh K. Sitaraman. His continuous encouragement, guidance, and confidence in me made this work possible. He gave me the primary motivation and inspiration for this research as well as generous support. His broad knowledge and insight about the reliability of electronic packaging systems guided me from the start of this journey to the end.

I also give my thanks to my advisor, Dr. Russell S. Peak. His expertise in engineering information and industry applications was the major guidance in this work. His financial support from Shinko, Rockwell Collins, ISR, NIST, and NASA projects is also greatly appreciated.

I give my special thanks to Dr. Robert E. Fulton for giving me the opportunity to study at Geogia Tech. After his death, I'd realized that his existence itself provided great support to me.

I owe my thanks to other committee members, Drs. Christiaan Paredis, Raghuram V. Pucha, and C. P. Wong, for agreeing to serve on my committee and providing valuable comments.

I cherish the support and compassion given by my colleagues in the CASPaR Lab. (Jamie Ahmad, Dr. Shashikant Hegde, Karan Kacker, Kevin Klein, Kang Joon Lee, Sakethraman Mahalingam, Dr. Andrew Perkins, Krishna Tunga, and Jiantao Zheng), those in EIS Lab.(Manas Bajaj, Dr. Greg Mocko, Dr. Diego R. Tamburini, Miyako

Wilson, and Dr. Sai Zeng), and my Korean friends in Georgia Tech. (Dr. Haejin Choi, Dr. Dongjoo Lee, Dr. Hangil Chae, Dr. Hungsun Shon, Sungchul Joo, Sangil Lee, Seil Lee, and Dr. Hyungwook Park).

I offer my deepest thanks to my parents and my mother-in-law for their continuous encouragement, sacrifice, and love throughout this process.

I would like to dedicate this dissertation to my wife, Kyunghwa Lee, my elder daughter, Jungmin Kim, and my younger daughter, Joomin Kim.

I thank my Lord for everything.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

GLOSSARY

ABB	Analysis Building Block
APM	Analyzable Product Model
APs	Application Protocols
AS	Assembly Structure
BGA	Ball Grid Array
CAD	Computer-Aided Design
CAE	Computer-Aide Engineering
CASDfR	Computer-Aided System Design for Reliability
CBAM	Context-Based Analysis Model
CPU	Central Processing Unit
DF	Design Feature
ECU	Engine Control Unit
ENC	Expected Number of Components
ECSCC	Expected Complexity Sum of Critical Components
FID	Failure Interaction and Dependency
FIT	Failure In Time
FM	Failure Mode
FMEA	Failure Modes and Effects Analysis
FMS	Failure Mode Structure
FTA	Fault Tree Analysis
LS	Logical Structure
MRA	Multi-Representation Architecture
MTBF	Mean Time between Failures
PC	Personal Computer
PTH	Plated Through Hole
PWB	Printed Wiring Board
PWBA	Printed Wiring Board Assembly
RA	Reliability Activity
RBD	Reliability Block Diagram
RfDC	Rules for Design Change
RFR	Random Failure Rate
RM	Reliability Metric

ROM	Reliability Object Model
ROM-Library	Reliability Object Model Library
ROM-Tree	Reliability Object Model Tree
RPM	Reliability Prediction Model
RTRW	Relative Target Reliability Weight
SDfR	System Design for Reliability
SMM	Solution Method Model
STEP	The Standard for the Exchange of Product Model Data
TENC	Total Expected Number of Components
TECSCC	Total Expected Compexity Sum of Critical Components
UC	Usage Condition
UML	Unified Modeling Language
USB	Universal Serial Bus
XML	Extensible Markup Language

LIST OF SYMBOLS

E	Young's Modulus
N_{50}	Number of Cycles to 50% Failure
$N_{50,hours}$	N_{50}×Frequency
$R(t)$	Reliability
$R_i(t)$	Item Reliability
$R_{i,j}(t)$	Sub-Item Reliability
$R_r(t)$	Random Failure Reliability
$R_{r,i}(t)$	Item Random Failure Reliability
$R_{r,i,j}(t)$	Sub-Item Random Failure Reliability
$R_w(t)$	Wearout Failure Reliability
$R_{w,i}(t)$	Item Wearout Failure Reliability
$R_{w,i,j}(t)$	Sub-Item Wearout Failure Reliability
t	Time
T_w	Time to Wearout Failure
w	Reliability Weight
w_a	Weight for Expected Number of Component Factor
w_b	Weight for Expected Sum of Complexity of Critical Component Factor
$w_{i,j}$	Sub-Item Reliability Weight
α	Characteristic Lifetime of Weibull Curve
β	Shape Parameter of Weibull Curve
ε	Strain
γ	Shear Strain
λ	Random Failure Rate
λ_i	Item Random Failure Rate
$\lambda_{i,j}$	Sub-Item Random Failure Rate
σ	Stress

SUMMARY

Microelectronic systems such as cell phones, computers, consumer electronics, and implantable medical devices consist of subsystems which in turn consist of other subsystems and components. When such systems are designed, fabricated, assembled, and tested, they need to meet reliability, cost, performance, and other targets for being competitive. The design of reliable electronic packaging systems in a systematic and timely manner requires a consistent and unified method for allocating, predicting, and assessing reliability and for recommending design changes at the component and system level with consideration of both random and wearout failures.

Accordingly, this dissertation presents a new unified knowledge modeling method for System Design for Reliability (SDfR) called the Reliability Object Model (ROM) method. The ROM method consistently addresses both reliability allocation and assessment for systems composed of series and parallel subsystems. The effectiveness of the ROM method has been demonstrated for allocating, predicting, and assessing reliability, and the results show that ROM is more effective compared to existing methods, providing richer semantics, unified techniques, and improved SDfR quality. Furthermore, this dissertation develops representative reliability metrics for random and wearout failures, and incorporates such metrics into ROM together with representative algorithms for allocation, assessment, and design change recommendations. Finally, this research implemented the ROM method in a computing framework and demonstrated its applicability using several relevant microelectronic system test cases and prototype SDfR tools.

CHAPTER 1

INTRODUCTION

This chapter introduces problems related to the reliability of complex electronic packaging systems. Then, it describes the motivation and the outline of this thesis.

1.1 COMPLEX SYSTEM FAILURES

Most systems are designed to perform required functions under stated conditions for a stated period of time [Bajenescu and Bazu, 1999]. However, systems fail unexpectedly in real life, and some system failures can have a range of results, from catastrophe to lost market share.

Generally, system failures are complex. For example, Intel's Pentium III chips caused system errors (Figure 1.1) when running certain programs and at a certain temperature [Fried, 2000]. Although individual components met their reliability targets, the integrated system failed at a given temperature while running certain programs. This type of failure, which differs markedly from intrinsic component failures, is very hard to detect and difficult to manage.

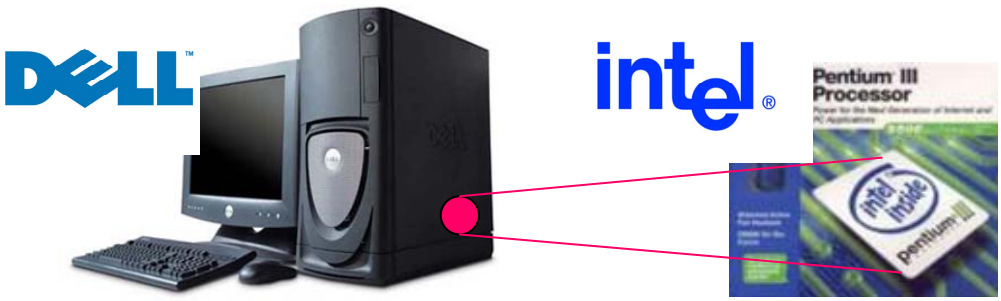


Figure 1.1 An example of system context-based failures
(The recall of Intel Pentium Ⅲ processor)

Another example is the random failure of Universal Serial Bus (USB) ports in a notebook PC, shown in Figure 1.2. Unlike wearout failures, random failures are sudden and often cannot be explained. To account for such random failures, some systems will need to include redundant components or subsystems.

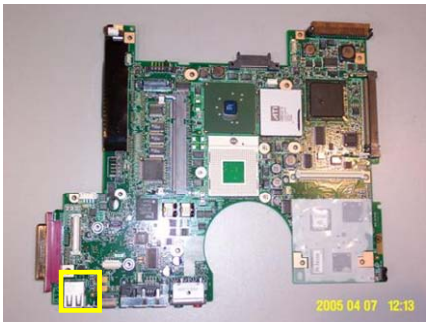


Figure 1.2 An example of random failures
(The sudden failure of USB ports in IBM note PC T42p)

In addition, interactions among highly integrated electronic components may cause system failures [Pucha et al., 2004]. For example, a high temperature-induced failure may be due to the presence of two or more high power packages assembled in close proximity to each other.

1.2 RELIABILITY OF COMPLEX SYSTEMS

One well-known measure of failures in engineering is "reliability." Reliability is the probability that an item operating under stated conditions will function as expected for a stated period of time [Jensen, 1995]. Reliability is an important factor in system development because high reliability typically makes systems competitive in the market and saves maintenance cost.

Studies of system reliability initially applied the stochastic knowledge of field failure data in the 1950s [Denson, 1998]. Field data-based research[1] addresses random failures during the useful lifetime of systems [Jensen, 1995]. Representative field data-based reliability prediction models are MIL-HDBK-217 and Telcordia SR-332 [Foucher et al., 2002]. These models facilitate reliability prediction of complex electronic systems. However, since they cannot explain the cause of failures, their ability to enhance reliability during system design phases is limited.

On the other hand, physics-based research, initiated in the 1960s [Ebel, 1998], deals with wearout failures and attempts to identify factors related to lifetime. Physics-based studies develop various accelerated testing and reliability simulation models (e.g. the accelerated thermo-cycle testing [Crowe and Feinberg, 2001] and the solder joint fatigue model [Engelmaier, 1983]). These physics-based models are useful in determining component design parameters and improving component reliability. However, they are limited when it comes to considering relations, interactions, and dependencies among components in systems [Denson, 1998; Snook et al., 2003].

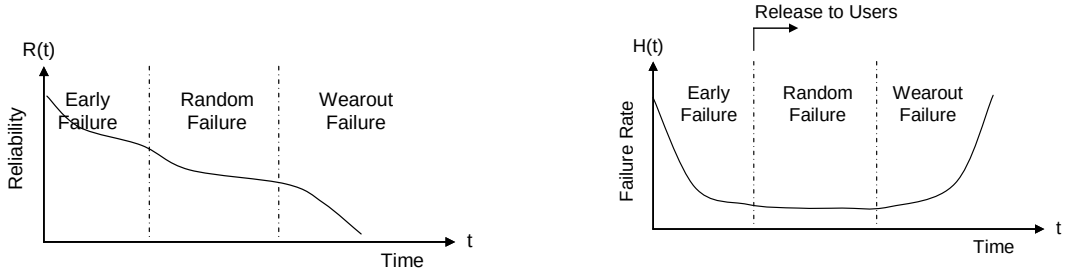
[1] Field data-based research is also called statistics-based research in this thesis.

The different aspects of field data-based and physics-based research are illustrated well by the bath-tub curve, a commonly accepted curve of hazard functions [Jensen, 1995]. This curve shows three distinct stages (Figure 1.3): an early failure stage, a random failure stage, and a wear-out failure stage.

The first stage of the bath-tub curve is characterized by early failures, also known as infant failures. Decreasing failure rate model can be used to describe infant failures (Equation (1.1)). These failures are typically caused by manufacturing errors, and systems with such potential infant mortality are often screened out and removed in the "burn-in" process. Systems that successfully pass through the burn-in process will be released for broad usage.

The second stage is characterized by random failures, caused by randomly changing operating conditions such as freak loads. These failures tend to occur at a constant rate (Equation (1.2)). Therefore, constant failure rate model can be used to describe random failures.

The last stage is characterized by wearout failures, which is caused by cyclic stress, mechanical wear, chemical reaction, and so on. These failures typically occur at an increasing rate. Therefore, increasing failure rate model can be used to describe wearout failures (Equation (1.3)).



(a) Reliability function (b) Hazard function of the bath-tub curve

Figure 1.3 Reliability and hazard functions

$$R\ (t) = e^{-(\frac{t}{\alpha})^{\beta}} \tag{1.1}$$

where

α: characteristic lifetime,

β: shape factor (β <1).

$$R(t) = e^{-\lambda t} \tag{1.2}$$

where

λ: constant random failure rate.

$$R\ (t) = e^{-(\frac{t}{\alpha})^{\beta}} \tag{1.3}$$

where

α: characteristic lifetime,

β: shape factor (β >1).

The lifetime characteristics of components often differ depending on the types of components. In general, for capacitors and transistors, failures that are seen in operation are usually random failures, not wearout failures. However, solder joints and plated through holes (PTH) fail by wearout. Since various types of components exist in a system, both the field data-based prediction method for random failures and the physics-based prediction method for wearout failures are necessary for system development [Condra, 1993; Foucher et al., 2002; Jensen, 1995]. The way of integrating two different methods may be a problem.

Another problem is analyzing the reliability of complex systems that consist of complex assembly structure with numerous components. Since a system failure can be caused by the failure of any component in an assembly of hundreds and thousands of

components, each component must be analyzed, one by one, following assembly structures and logical structures. It takes a lot of time and effort.

Furthermore, analyzing multiple failure modes and mechanisms of systems is also a problem. For example, in an assembly system, a solder joint failure may be different when subjected to thermal cycling versus mechanical vibration versus a combination of both. Therefore it can take significant effort to isolate and analyze such different failure modes and mechanisms.

Similarly, in an electronic system design, one needs to be cognizant of mechanical-induced failures (creep, delamination, etc.), electrical-induced failures (electro-migration, junction spiking, etc.), and chemical-induced failures (corrosion, diffusion, etc.) [Tummala, 2001]. Therefore, it is necessary for different design groups with different domain expertise to work together.

1.3 MOTIVATIONS

The ultimate goal of such reliability studies is to enhance system reliability. Most current system reliability enhancement methods are applied at the prototype stage of the system development lifecycle using accelerated testing methods. Although they are effective reliability enhancement methods, they are expensive because any unsatisfactory results of reliability testing require redesign and retest cycles until the results are satisfactory.

If reliability knowledge is applied during the design stage, the chances of redesign and retest cycles may be reduced. Therefore, it may be a cost-effective reliability enhancement method. The approach of considering reliability at the design stage is

referred to as “design-for-reliability,” [Crowe and Feinberg, 2001] which increases the chance of developing reliable designs and decreases the burden of downstream reliability activities such as reliability testing, warranty service, maintenance, and analysis of field failures (Figure 1.4).

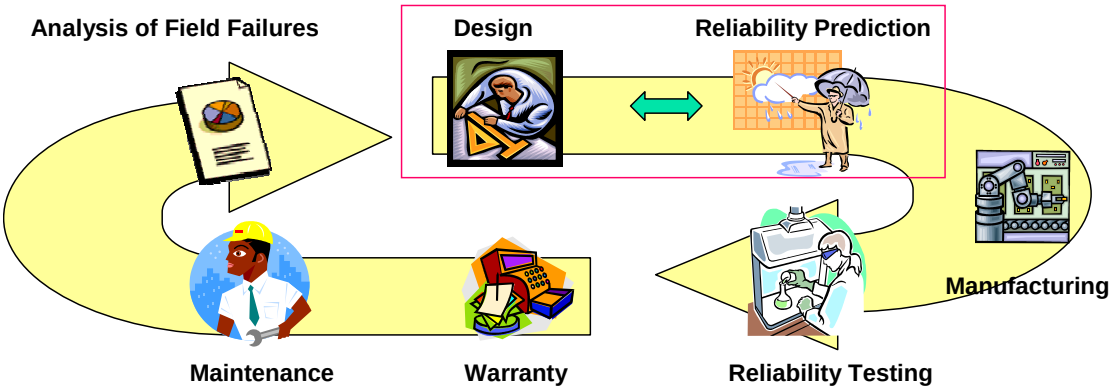


Figure 1.4 Design-for-Reliability

Most current design-for-reliability studies focus mainly on component design, not on system design. This is due in part to difficulties in sharing reliability goals among multiple subsystem design groups and applying diverse reliability prediction and analysis knowledge to complex system structures and design processes. Therefore, it is necessary to develop a method to assess the system-level reliability to meet reliability targets for current and future microelectronic systems.

1.4 THESIS OUTLINE

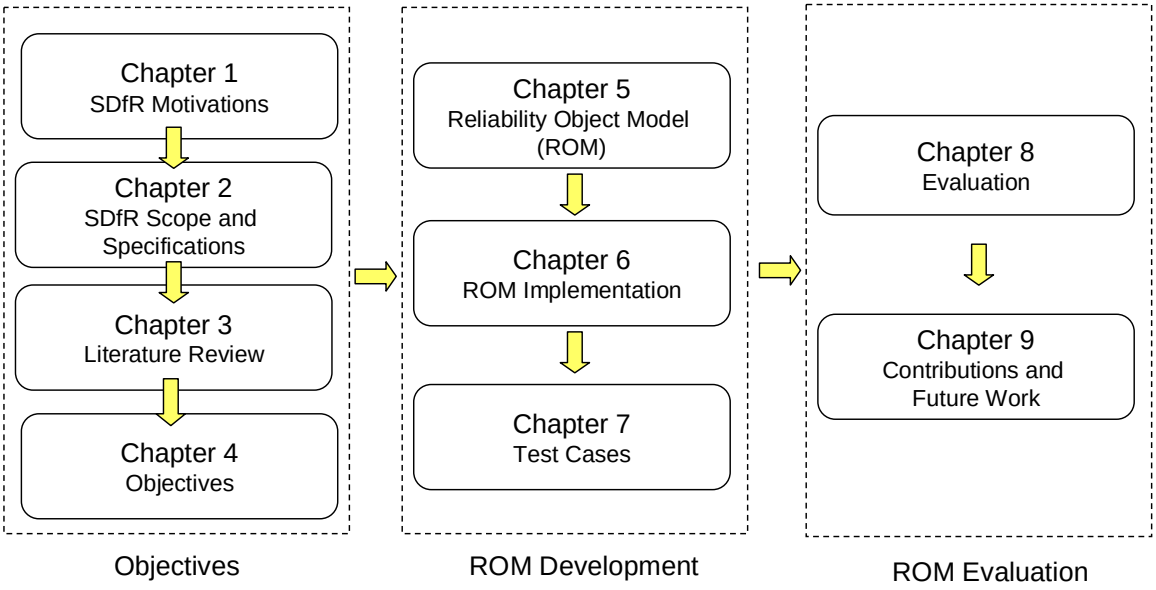


Figure 1.5 Thesis structure

The outline of this thesis is illustrated in Figure 1.5. In Chapter 2, the scope and the specifications of system design for reliability (SDfR) are described. Chapter 3 reviews on the existing studies related to SDfR. Chapter 4 summarizes the research gaps in existing literature and outlines the thesis objectives.

Chapter 5 describes a knowledge model for SDfR, referred to as Reliability Object Model (ROM), which includes ROM metrics, ROM algorithms, and ROM-Structure, ROM-Library, and ROM-Tree. Chapter 6 describes an implementation of ROM. This implementation includes an information framework for SDfR and prototype computer-aided SDfR (CASDfR) tools. For the demonstration of the ROM method, Chaper 7 presents four test cases: 1) a video broadcasting system test case for target random failure reliability allocation 2) a notebook PC test case for target wearout failure reliability allocation, 3) a Universal Serial Bus (USB) hub test case for reliability

prediction and assessment, and 4) an Engine Control Unit (ECU) test case for design recommendations.

Chapter 8 discusses the evaluation of the ROM method with respect to the research objectives. Chapter 9 discusses the contributions of this research and future work.

CHAPTER 2

SDFR SCOPE AND SPECIFICATIONS

We have defined system design for reliability (SDfR) as follows [Kim et al., 2005]:

> ***System Design for Reliability (SDfR)*** *is a method that is used for designing subsystems, selecting components, and designing interconnections to satisfy target system reliability in a given context.*

Based on the SDfR definition, this chapter describes SDfR scope and provides specifications for in-depth study.

2.1 SDFR SCOPE

2.1.1 Application Aspect

A "system" is defined as a collection of components organized to accomplish a specific function or a set of functions [IEEE, 1990]. Most complex systems include both software components and hardware components. The scope of this research is limited to electronic hardware. Therefore, in this research, "system" refers to the assembly of electronic hardware.

2.1.2 System Design Aspect

For system design, requirements are developed at the beginning and then transformed to engineering functionalities. Based on the functionalities, hardware

modularization is done, and multiple design groups work concurrently to realize the assigned modules selecting components and designing their interconnections. Then, embodied modules are integrated and verified with regards to the associated requirements.

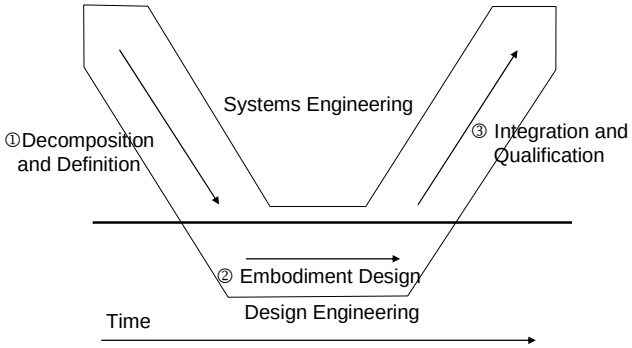


Figure 2.1 The Vee model

These system design activities are well described in the Vee model [Buede, 2000] that represents the Systems Engineering process. This is illustrated in Figure 2.1. The Vee model is comprised of two parts: design engineering and systems engineering. The Systems Engineering portion of the Vee model is described by two axes, the decomposition of the system to design engineering and the integration of design engineering to the system level.

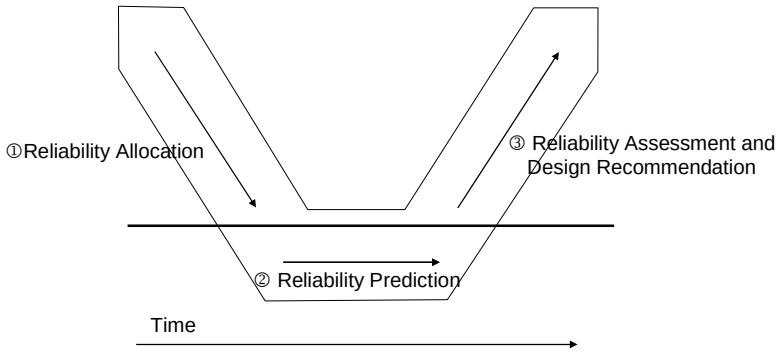


Figure 2.2 SDfR activities laid out on the Vee model

Based on the Vee model, we have arranged reliability activities for system design (Figure 2.2): reliability allocation; reliability prediction; and reliability assessment and design recommendation. Reliability allocation is the reliability aspect of system decomposition and definition. Reliability prediction is the reliability aspect of embodiment design. Reliability assessment and design recommendation is the reliability aspect of system integration and qualification.

From the Systems Engineering process, system design characteristics are distinguished compared with component design (Table 2.1). One system design characteristic is that systems are designed from requirements from specific system usages. For example, the temperature requirement for cell phones depends on where human lives. Therefore, the requirement might be set from -5°C to 50°C. However, the temperature requirement for transistors is set based on the general transistor usages or standards, not for specific usages.

Table 2.1 Characteristics of system design

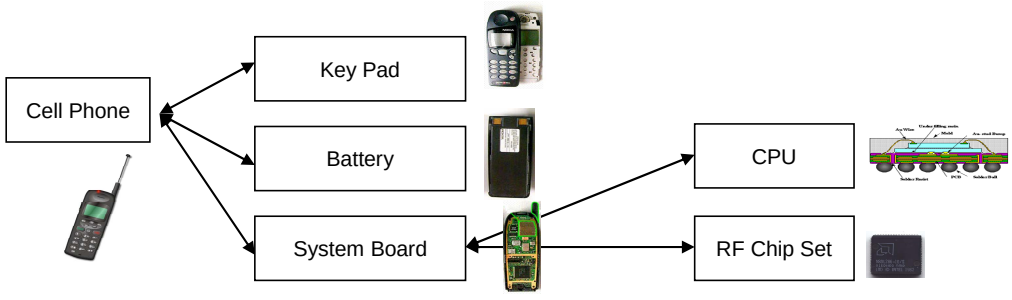
Component Design	System Design
e.g. Transistor	e.g. Cell phone
• Requirements from general component usages • Uni-functionality • One design group • Uni-level manufacturing structure • Material selection and shape design	• Requirements from specific system usages • Multi-functionalities • Multiple design groups • Multi-level assembly structure • Component selection and interconnection design

Another characteristic is that systems are supposed to perform multiple functions rather than one function. Therefore, system failures should be understood based on multiple functionalities. For example, cell phones perform two main functions: calling

and answering. Therefore, cell phones may fail in calling, answering, or both of them. In this research, we do not specify system failures based on each function. We only consider system failures as overall main functionality failures. Decomposing failures with respect to system functions will be future work.

The involvement of multiple design groups is another characteristic of system design because of multi-level and multi-disciplinary nature of systems, as shown in Figure 2.3. Figure 2.3-(a) shows an example multi-level system design of cell phone design. *Cell Phone* design is decomposed into *Key Pad*, *Battery*, and *System Board* design. Then, *System Board* design is decomposed into various chip package design such as *CPU* and *RF Chip Set* design. Figure 2.3-(b) shows an example multi-disciplinary system design of hard disk design. *Hard Disk* design is decomposed into *Electronic System* and *Mechanical System* design.

In general, multi-level and multi-disciplinary design causes miscommunication and incomplete knowledge sharing problems among multiple design groups, and these problems are also valid for SDfR.



(a) Multi-level system design

Figure 2.3 Multi-level and multi-disciplinary system design (Continue)

Electronic System

Mechanical System

(b) Multi-disciplinary system design

Figure 2.3 Multi-level and multi-disciplinary system design

The last characteristic is that system design includes a multi-level design problem. Therefore, a system design method integrates various design activities such as system configuration design, component selection, interconnection design, and material selection and design.

2.1.3 System Reliability Aspect

To design a reliable system, one must consider the six general causes of system failures (Figure 2.4): 1) intrinsic failures of components, 2) intrinsic failures of interconnections, 3) field usage conditions 4) interactions among components, 5) manufacturing errors, and 6) user errors. Generally speaking, system failures occur owing to some combinations of these like the example of the recall of Intel Pentium Ⅲ processor shown in Figure 1.1.

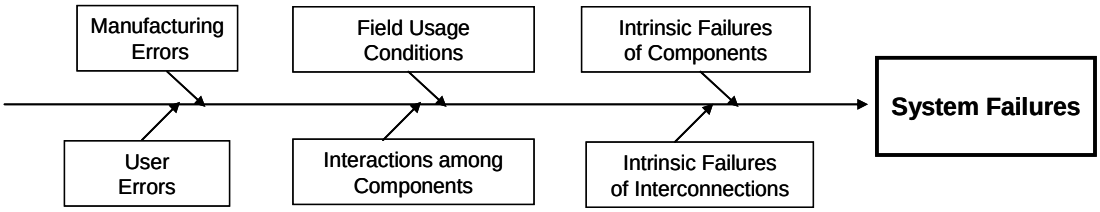


Figure 2.4 A causes and effects diagram of system failures

The intrinsic failures of components and interconnections are predicted using field data-based reliability prediction models and physics-based reliability prediction models. The field data-based models are for random failures, and the physics-based models are for wearout failures.

The random failure reliability and the wearout failure reliability are predicted under specific system field usage conditions. Then, the prediction results are integrated for assessing system reliability following three types of structures: 1) physical assembly structure, 2) logical structure, and 3) failure mode structure. Physical assembly structure is a set of organized relations among systems, subsystems, and components. Logical structure is a set of organized relations among series and parallel connections of physical items in terms of possible failures. Failure mode structure is a set of organized relations among failure modes and mechanisms.

Beside these structures, interactions in terms of failures among components are also important factors for assessing system reliability. These make system reliability analysis and assessment complex.

This research has disregarded manufacturing errors and user errors because the quantification and the consideration of them at the design stage are difficult and limited.

2.1.4 Summary

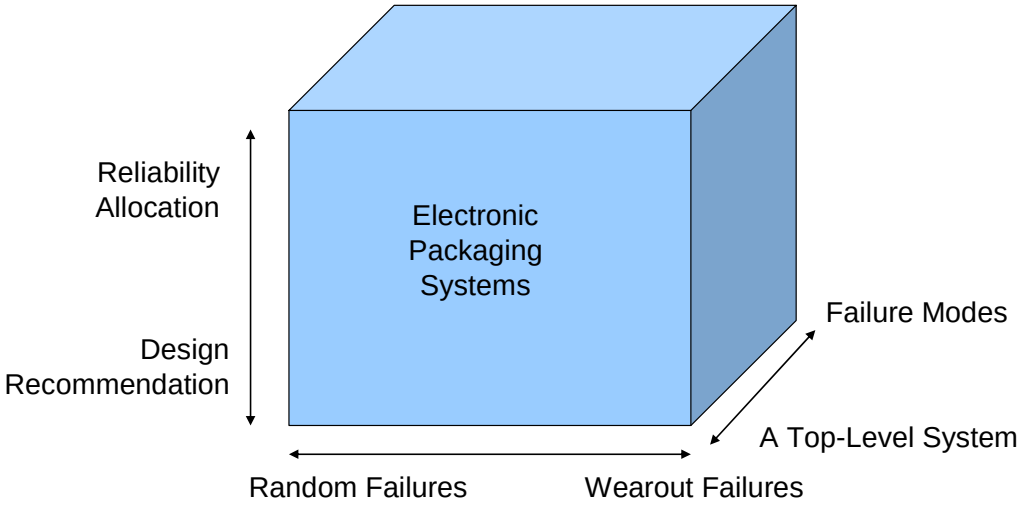


Figure 2.5 SDfR Scope

In summary, SDfR covers concepts ranging from reliability allocation to design recommendations, from random failures to wearout failures, and from a top-level system to failure modes for designing electronic packaging systems. The SDfR scope is illustrated in Figure 2.5.

2.2 SDFR SPECIFICATIONS

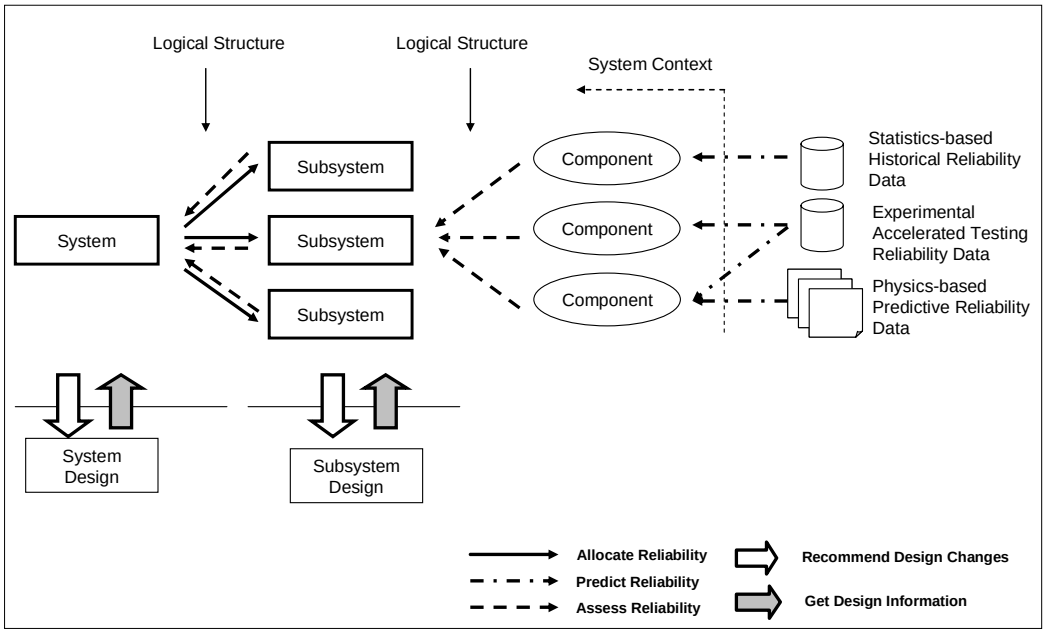


Figure 2.6 A conceptual model for SDfR

Within the scope, we have developed a conceptual model for SDfR, illustrated in Figure 2.6. First, target reliability is allocated from parent systems to subsystems, as illustrated using solid lines in Figure 2.6. The allocated subsystem target reliability is then subsequently used to determine the target reliability of its subsystems, and so on. This target reliability allocation activity is top-down in nature. In current practice, it requires assembly structures[2], logical structures[3], and target reliability metrics as input information.

Second, the reliability of components or failure modes are determined from statistics-based field data or physics-based lifetime models, as illustrated using dash-dot

[2] An assembly structure is a set of organized relations among systems, subsystems, and components.
[3] A logical structure is a set of organized relations among series and parallel connections of physical items in terms of possible failures.

lines in Figure 2.6. The reliability prediction activity requires design features, reliability prediction models, and usage conditions as input information.

Third, overall system reliability is then assessed from components to subsystems and from subsystems to parent systems, in a bottom-up fashion as illustrated by dashed lines in Figure 2.6. This reliability assessment activity requires assembly structures, logical structures, failure mode structures[4], failure interactions and dependencies[5], and predicted reliability metrics as input information.

Fourth, when the assessed reliability is greater than or equal to the assigned target reliability, then no design modification is recommended. However, when the assessed reliability is less than the assigned target reliability, design changes are recommended, as illustrated using boxed arrows in Figure 2.6. When subsystem design changes are not possible or available, design changes may be initiated at the parent system level so that redundant, modularized, or alternative subsystems may be pursued. The design change recommendation activity requires rules for design changes, target reliability metrics, and assessed reliability metrics as input information.

The reliability activities (RA) in Figure 2.6 are represented in functional form, in Table 2.2. From these functional forms, the nine SDfR specifications (Table 2.3) are identified. These are: 1) assembly structures (AS), 2) design features (DF), 3) failure interactions and dependencies (FID), 4) failure mode structures (FMS), 5) logical structures (LS), 6) rules for design changes (RfDC), 7) reliability metrics (RM), 8) reliability prediction models (RPM), and 9) usage conditions (UC).

[4] A failure mode structure is a set of organized relations among failure modes and mechanisms.
[5] Failure interactions and dependencies are relations in terms of failures between two items.

Table 2.2 Reliability activities

Reliability Allocation	$RM_{target}^{subsystem} = RA_{allocate}(AS, LS, RM_{target}^{system})$
Reliability Prediction	$RM_{predicted}^{component\ /\ failure\ mode} = RA_{predict}(DF, RPM, UC)$
Reliability Assessment	$RM_{assessed}^{subsystem\ /\ system} = RA_{assess}(AS, FID, FMS, LS, RM_{predicted}^{component\ /\ subsystem})$
Design Recommendations	$(AS, DF, LS)_{candidate\ changes}$ $= RA_{recommend}(RfDC, RM_{target}^{system/subsystem}, RM_{assessed}^{system/subsystem})$

Table 2.3 SDfR specifications

AS:	Assembly Structures	RfDC:	Rules for Design Changes
DF:	Design Features	RM:	Reliability Metrics
FID:	Failure Interactions & Dependencies	RPM:	Reliability Prediction Models
FMS:	Failure Mode Structures	UC:	Usage Conditions
LS:	Logical Structures		

2.3 DISCUSSION

The conceptual model illustrated in Figure 2.6 leads to design decisions at the subsystem level. This approach assumes that component redundancies are dominant over subsystem redundancies. Theoretically, systems that consist of component redundancies are more reliable than system that consists of subsystem redundancies [Leemis, 1995]. For example, suppose two systems consist of the same components but have different constructions for redundancy. The columns in Figure 2.7 present such systems where circles represent components and squares represent subsystems (Figure 2.7). One consists of component redundancies (Figure 2.7-(a)) in which each component is backed

up by its own redundant component that is represented by a dashed circle (e.g. C_{a1} and C_{b1}). The other consists of a subsystem redundancy (Figure 2.7-(b)) in which subsystem $S_{1.1}$ is backed up by an identical subsystem $S_{1.2}$ that is represented by a dashed square. As shown in block diagram views in Figure 2.7, suppose components C_{a2} and C_{bn} in Figure 2.7 fail in both *System A* (S_A) and *System B* (S_B). Then, S_A still runs fine, but S_B fails because of the different constructions for redundancy. The effectiveness of component redundancy is also proved mathematically by the inequality equation [Leemis, 1995] in Figure 2.7, where R(t) represents reliability. Therefore, we believe the approach in Figure 2.6 is effective for SDfR.

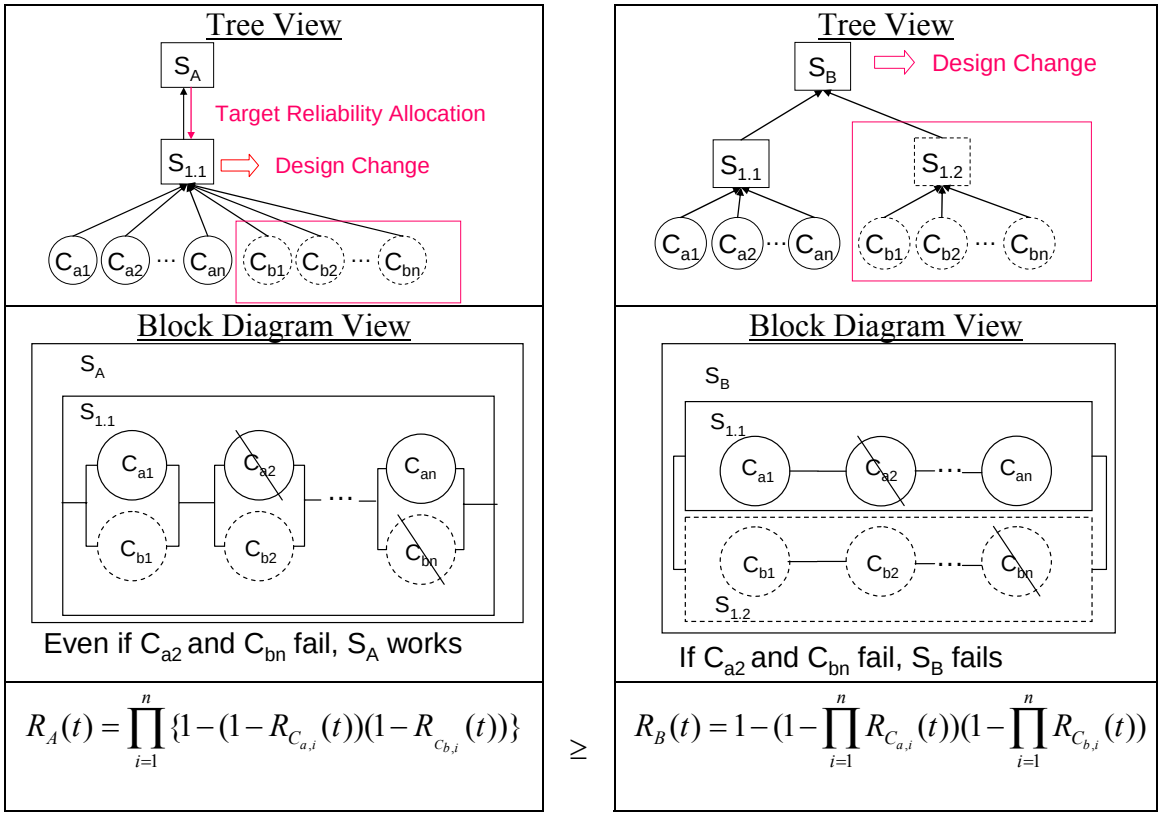


Figure 2.7 Comparison between component redundancy and subsystem redundancy

CHAPTER 3

LITERATURE REVIEW

To understand the current status of system-level reliability, this chapter presents a review of existing literature. The research areas reviewed include reliability metrics, reliability allocation methods, reliability prediction methods, reliability analysis and assessment methods, reliability enhancement methods, and reliability of electronic packaging.

3.1 RELIABILITY METRICS

According to Wood [Wood, 2001], the definition of a failure may vary according to person's perspective. To a hardware engineer, a failure means part replacement and verification of the failure. To a service organization, failure is a service call for corrective maintenance. To a customer, a failure is a downgrade of the product capability. These various definitions of failure lead to various types of reliability metrics, such as constant random failure rate [DoD, 1991], part replacement rate, service call rate, mean-time-between-failure (MTBF), percentile lifetime [Jensen, 1995], and so on [Wood, 2001].

Wood [Wood, 2001] grouped reliability metrics into two general categories of metrics: constant rate metrics and probability of success metrics. Constant rate reliability metrics, which are based on exponential distribution, are widely used in electronic industries. This is because they are good approximations of the reliability behavior of a

product during its useful life and simple to calculate [Wood, 2001]. For example, if the constant failure rate is represented by the parameter λ, λ = 2000 FIT means around two failures out of 100 items in a year. FIT is failure in time, commonly number of failures per billion hours of usage.

Various constant rate reliability metrics are listed in Table 3.1. These constant rate reliability metrics represent random failures well, but not wearout failures because the failure rate is not constant in wearout failure region. For wearout failures, probabilities of success metrics that are based on Weibull distribution are commonly used. Percentile life time is a typical probability of success metrics [Jensen, 1995]. For example R96C90 in automotive industry means a reliability of 96% with 90% confidence [Wood, 2001]. Another probability of success metrics is the number of cycles at x% failures. For example, N_{50} means number of cycles to 50% failures [Dowling, 1999; Rao, 1992].

Table 3.1 Constant rate reliability metrics [Wood, 2001]

Constant Rate Metric	Mean Life Equivalent	Definition
Failure rate	MTBF*	Total failures divided by total population operating time
Part return rate	MTBPR* (R=return)	Total parts returned divided by total population operating time
Part replacement rate	MTBPR* (R=replacement)	Total parts replaced divided by total population operating time
Service call rate	MTBSC*	Total service calls divided by total population operating time
Warranty claim rate	MTBWC*	Total warranty claims divided by warranted population operating time
Service interruption rate	MTBSI*	Total service interruptions divided by total population operating time
Maintenance action rate	MTBMA*	Total maintenance actions divided by total population operating time
* MTBX means Mean-Time-Between-X		

As Wood asserted that multiple reliability metrics are required for supporting system lifecycle [Wood, 2001], one reliability metric is not enough for supporting SDfR that covers from random failures to wearout failures. Therefore, to define effective reliability metrics for SDfR is necessary, so that SDfR activities are carried out successfully.

3.2 RELIABILITY ALLOCATION METHODS

Kapur and Lamberson [Kapur and Lamberson, 1977] give two key reasons for the use of reliability allocation in designing systems as follows:

(1) It forces the designer to understand and develop the relationship between component, subsystem, and system reliabilities.

(2) The designer can consider reliability in a framework that incorporates other issues such as cost, physical dimensions, weight, etc. in the design process.

Thus, reliability allocation is necessary.

If there is established objective function such as cost or performance, optimal solutions for reliability allocation exist. Such optimization methods are studies by [Dhingra, 1992; Elegbede et al., 2003; Kartik and Murthy, 1995; Kuo and Prasad, 2000; Mettas, 2000; Painton and Campbell, 1995; Park, 1987]. If there is no established objective function, the only solution or the best solution does not exist. Therefore, various heuristic or strategic methods have been suggested depending on different context. Table 3.2 shows existing heuristic reliability allocation methods.

Table 3.2 Reliability allocation methods [Blischke and Murthy, 2000; Falcone et al., 2002]

Heuristic Allocation Methods	Description
Equal Apportionment Method	When all subsystems are identical, subsystem reliability weights are equally distributed.
ARINC Method	This method is proposed by Aeronautical Research Inc. Subsystem reliability weights are determined based on historical data.
BOYD Method	This method is a factor sum of the Equal Apportionment method and the ARINC method.
AGREE Method	This is proposed by the Advisory Group on Reliability of Electronic Equipment, Office of the Assistant Secretary of Defense. It takes into consideration the complexity and importance of each subsystem.
KARMIOL Method – factors product	This method considers the influence of different four factors: (1) Complexity (2) State of Art-Technology (3) Operative profile (4) Criticality The product of the four factors represents the reliability weight for each subsystem.
KARMIOL Method – factors sum	This method considers the influence of different four factors: (5) Complexity (6) State of Art-Technology (7) Operative profile (8) Criticality These factors are summed to obtain the total weight factor for each subsystem.

However, these heuristic reliability allocation methods still includes two main issues: what to allocate and how to allocate. With regard to what to allocate, most existing research focus on allocating only target reliability metrics for constant random failures [Blischke and Murthy, 2000; Dhingra, 1992; Park, 1987], not target reliability metrics for wearout failures. This is because of the mathematical difficulty in handling multiple parameters that represent wearout failures.

For the second issue of how to allocate, existing reliability allocation methods only address series connections [Blischke and Murthy, 2000], but not complex connections that are combinations of series and parallel structures. This is because of mathematical complexity.

In summary, the existing reliability allocation methods in Table 3.2 are limited for SDfR because SDfR covers both random and wearout failures and considers complex systems that include series and parallel structures.

3.3 RELIABILITY PREDICTION METHODS

Reliability prediction is important for SDfR because it helps performing trade-off studies, planning for design improvement, and providing a basis for warranty and maintenance plans [Healy et al., 1997].

For reliability predictions at the early system development stage, three typical types of reliability prediction methods are available: field data-based methods, accelerated testing-based methods, and physics-based methods.

3.3.1 Field Data-based Reliability Prediction Methods

In the 1950's, the poor intrinsic quality and reliability of components often caused electronic system failures, so the studies on system reliability focused on analyzing constant random failure rates of components from field data [Denson, 1998]. Random failure rates are assumed constant during the useful lifetime of systems based on the concept of the bath-tub curve, and the assumption simplifies the calculation of system

reliability [Jensen, 1995]. Table 3.3 lists the most common field data-based reliability prediction models and their latest update.

Table 3.3 Field data-based reliability prediction models [Foucher et al., 2002]

SAE reliability prediction method	1987	HRD-5	
Mil-Hdbk-217	1995	Siemens SN29500	1999
Telcordia SR-332 (Bellcore)	1997	NTT procedure	1985
CNET RDF-93	1993	Reliability Analysis Center	2000
CNET RDF-2000	2000	PRISM	
British Telecom	1995		

Even though, the field data-based reliability analysis reflects actual field data and is useful at the conceptual and logical design phases [Denson, 1998], it has some drawbacks. One drawback is that collecting and maintaining up-to-date field data are difficult. Another is that the causes of failures cannot be explained, so preventing them by design changes is limited.

3.3.2 Accelerated Testing-based Reliability Prediction Methods

When field reliability data are not available, another way of predicting reliability is accelerated testing, which compresses time and accelerates the failure mechanisms during a reasonable test period [Crowe and Feinberg, 2001]. The concept of accelerated testing is based on the Arrhenius reaction formula, Equation (3.1) [Crowe and Feinberg, 2001; Jensen, 1995]. The reaction rate in Equation (3.1) is assumed to be inversely proportional to the time that will occur, so the increase in the rate caused by an increase in temperature will reduce the reaction time., shown in Equation (3.2) [Crowe and Feinberg, 2001; Jensen, 1995].

$$r = A \cdot \exp[\frac{-E_A}{KT}] \tag{3.1}$$

where

r: the reaction rate,

T: the absolute temperature in degrees Kelvin,

K: the Boltzmann's constant ($8.6173 \times 10^{-5} eV/°K$),

E_A: the activation energy,

A: the frequency factor.

Equation (3.2) is valid only in the condition of linearity between two different temperatures and a large enough sample size [Crowe and Feinberg, 2001]. Although it is a credible assessment of reliability, accelerated testing is very expensive because any unsatisfactory results of such testing require redesign and retest cycles until the results are satisfactory.

$$\frac{t_2}{t_1} = \exp[\frac{E_A}{K}(\frac{1}{T_2} - \frac{1}{T_1})] \tag{3.2}$$

where

t_1: the reaction time at temperature T_1,

t_2: the reaction time at temperature T_2.

3.3.3 Physics-based Reliability Prediction Methods

Another way of predicting reliability at the design stage is to apply physics-of-failure knowledge. Studies of physics-based reliability began as an important aspect of

the reliability field in the early 1960s [Ebel, 1998]. As a result of these studies, most failure mechanisms were identified, and a large number of design guidelines, formula, and reliability prediction models were suggested. Table 3.4 lists the most common failure mechanisms of electronic components [Snook et al., 2003; Tummala, 2001], and Figure 3.1 shows some typical physics-based lifetime prediction models for electronic package systems [Engelmaier, 1983; Mercado-Corujo, 2001; Tunga, 2004]. Solder joints experience fatigue load by shear stress caused by the different thermal expansion between the component and the board, and a PTH experiences fatigue load by normal stress caused by the different thermal expansion between the PTH and the board.

Table 3.4 Failure mechanisms of electronic components

Overstress Failures		**Wear out Failures**	
Mechanical	Yield Fracture Delamination	**Mechanical**	Fatigue Creep Wear Delamination
Electrical	EOS (electrical overstress) ESD (electrostatic discharge) EMI (electromagnetic interference) Dielectric breakdown	**Electrical**	Electro-migration Junction spiking Hillock formation
Chemical		**Chemical**	Corrosion Dendrites Diffusion

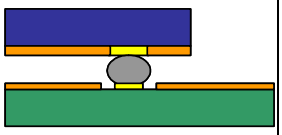
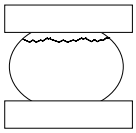
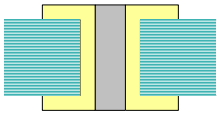
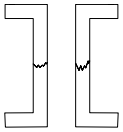
Component	Failure Mode	Lifetime Prediction Model
Solder Joints 	Thermo-mechanical Failure 	Engelmaier's Modified Coffin-Manson Equation for Solder $N_{50} = 0.5[\frac{\Delta\gamma_t}{2\varepsilon_f^{'}}]^{1/c}$
Plated Through Hole (PTH) 	Thermo-mechanical Failure 	Engelmaier's Modified Coffin-Manson Equation for Copper $\Delta\varepsilon_t = N_{50}^{-0.6}(\varepsilon_f^{'})^{0.75} + \frac{0.9\sigma_u}{E_p}[\frac{e^{\varepsilon_f^{'}}}{0.36}]^{0.1785}$

Figure 3.1 Physics-based lifetime prediction models

These physics-based models are useful in determining design parameters and improving reliability. However, they are limited in considering relations, interactions, and dependencies among components in systems [Denson, 1998; Snook et al., 2003].

3.3.4 Computer-Aided Reliability Prediction Tools

Table 3.5 Computer-aided reliability prediction tools

Field Data-Based Reliability Prediction Tools
Lambda Predict [ReliaSoft, 2006] Relex Reliability Prediction [Relex, 2006] Item Reliability Prediction [Item, 2006]
Physics-Based Reliability Prediction Tools
CalcePWA [Osterman and Stadterman, 1999] RAMPS [Ahmad and Sitaraman, 2002] RCAE [Janasak, 2001] CADMP-II [Evans et al., 1995]

Based on these reliability prediction models, computer-aided reliability prediction tools are developed for immediate prediction of reliability during iterative system design processes. Various reliability prediction tools are listed in Table 3.5. Since various types of failures occur when systems run, integrating various reliability prediction methods, models, and tools is necessary for SDfR.

3.4 RELIABILITY ANALYSIS AND ASSESSMENT METHODS

Since reliability activities are carried out based on the reliability analysis results system reliability analysis is essential. Existing reliability analysis methods are illustrated in Figure 3.2.

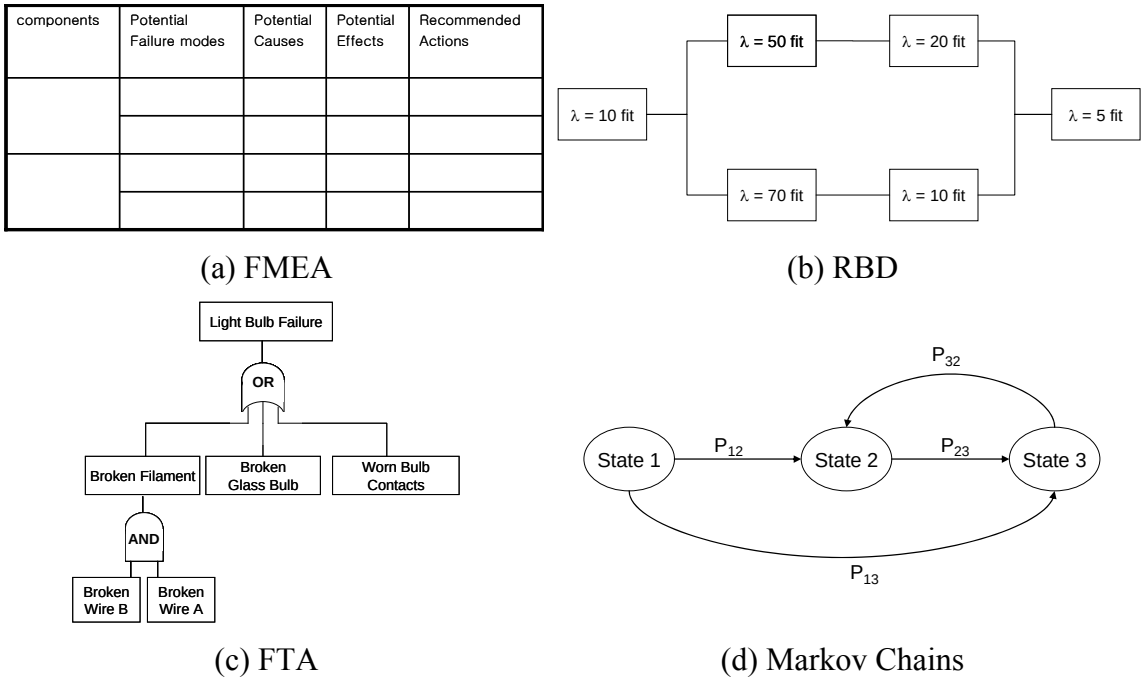


Figure 3.2 System reliability analysis methods

First, Failure Modes and Effects Analysis (FMEA) [Birolini, 2004; Blischke and Murthy, 2000; Crowe and Feinberg, 2001] is a method for analyzing potential failure modes early in the development cycle, thereby enhancing reliability through design. While anticipating every failure mode is not possible, the development team should formulate as extensive a list of potential failure modes as possible [Crow, 2002]. FMEA data can be collected in a table such as: *Component, Function, Failure Mode*, *Effect of Failure*, *Severity Rating*, *Potential Cause of Failure*, *Occurrence Rating*, *Possible Means*

of Detection, *Detection Rating*, *Risk Priority Number*, and *Preventative Actions to be Taken.* Even though FMEA is useful in system design, it is not able to discover complex system failures involving multiple components or subsystems.

Second, Reliability Block Diagram (RBD) [Birolini, 2004; Blischke and Murthy, 2000] is a component-based logic diagram that can simultaneously represent complex series and parallel structures of components. Boxes in RBD represent components, and lines in RBD represent series and parallel connections. The RBD can be used to carry out system reliability or steady state availability calculations [NORGE, 2007].

Third, Fault Tree Analysis (FTA) [Birolini, 2004; Blischke and Murthy, 2000] is an event-based logic diagram that displays the relationship between a potential event affecting system performance and an underlying cause for this event. Since the FTA is a top-down approach, it is possible to start the analysis at a very early stage and to complete it as the detailed design is carried out. The FTA is usually written out using conventional logic gate symbols such as the OR gate that represents series connection and the AND gate that represents parallel connection.

Fourth, Markov Chains [Birolini, 2004; Blischke and Murthy, 2000] is a sequence of random values whose probabilities at a time interval depend upon the value of the previous time [Carter, 1996]. Markov Chains can be used to analyze the processes or states of systems.

The characteristics of these reliability analysis methods are summarized with respect to basic SDfR specifications in Table 3.6. According to the table, each of these analysis methods partially supports basic SDfR specifications, but not fully [Amari, 2006]. FMEA supports only one level *Assembly Structure*, *Failure Mode Structure*, and

Reliability Metrics. However, it cannot support *Failure Interactions and Dependencies* and *Logical Structure*. RBD supports *Assembly Structure*, *Logical Structure*, and *Reliability Metrics*. However, it cannot support *Failure Interactions and Dependencies* and *Failure Mode Structure*. FTA supports *Assembly Structure*, *Logical Structure*, and *Reliability Metrics*. However, it cannot support *Failure Interactions and Dependencies* and *Failure Mode Structure*. Markov Chain supports only *Failure Interactions and Dependencies* and *Reliability Metrics*. However, it cannot support *Assembly Structure*, *Logical Structure*, and *Failure Modes Structure*.

Table 3.6 Basic SDfR specifications and existing reliability analysis methods

Basic SDfR Specifications	FMEA	RBD	FTA	Markov Chains
Assembly Structure	○	○	○	
Logical Structure		●	●	
Failure Mode Structure	○			
Failure Interactions and Dependencies				○
Quantitative Analysis (Reliability Metrics)	○	●	●	●

●: Strongly Supported, ○: Partially Supported, Blank: Not Supported.

Since each existing reliability analysis method cannot support complex systems fully, approaches of enhancing or combining existing methods have been studied. Vemuri et al. developed Reliability Imbedded Design Language (RIDL) to combine RBD and FTA [Vemuri et al., 1999]. Shalev et al. developed Condition-based FTA (CBFTA) method to update reliability values of a specific system and to calculate the residual life according to the system's monitored conditions [Shalev and Tiran, 2007]. Amari proposes a hybrid modeling method, which allows breaking up a system into several components, analyzing each component with the best suited reliability analysis methods,

and then combining the results together by information links [Amari, 2006]. Information tools that integrate existing methods also are surveyed such as IRIS [Jones et al., 2003] and RODON [SORMAN, 2006]. IRIS supports the exchange of reliability information generated by existing reliability analysis methods between the customer and the supplier in order to provide assurance that the product will meet the reliability requirements. RONDON is a model-based reasoning tool that supports engineers in their quest for reliable, well-designed technical product. It offers wide range of reliability analyses such as FMEA, FTA, or Diagnostic Decision Trees.

However, such approaches that are enhancing or combining existing methods have problems in efficiency. One problem is the load of maintaining the links and checking consistency among various analysis methods whenever designs are changed. Another is the unnecessary effort of keeping duplicated reliability information from various analysis methods. Therefore, a new unified method that analyzes system reliability from a top-level system to failure modes is needed.

Various reliability analysis tools are listed in Table 3.7. These tools are based on the existing reliability analysis methods: FTA, RBD, FMEA, Markov chains. Since a new reliability analysis method is necessary for SDfR, associated tools are also necessary for immediate system reliability analysis.

Table 3.7 Computer-aided reliability analysis tools

FTA and RBD Tools	**RBD Tools**
BlockSim FTI [ReliaSoft, 2006] Relex Fault Tree [Relex, 2006] ITEM Fault Tree [Item, 2006]	BlockSim [ReliaSoft, 2006] Relex RBD [Relex, 2006] ITEM RBD [Item, 2006]
FMEA Tools	**Markov chains**
XFMEA[ReliaSoft, 2006] Relex FMEA [Relex, 2006] ITEM FMECA [Item, 2006]	Relex Markov [Relex, 2006] Markov Module [Item, 2006]
Integrated Tools Supporting FTA, RBD, FMEA, and Markov chains	
Relex Reliability Studio [Relex, 2006] IRIS [Jones et al., 2003] RODON [SORMAN, 2006]	

3.5 RELIABILITY ENHANCEMENT METHODS

The ultimate goal of reliability studies is to enhance system reliability during the lifecycles by applying reliability knowledge. Nevertheless, when, where, and how to apply this knowledge for the effective enhancement of system reliability have become problematic. Attempts to solve such problems have led to the development of reliability programs such as MIL-STD-785 [Blischke and Murthy, 2000], ISO 9000 series [ISO, 2007], BS5760 [O'Connor, 2007], SAE M-100 [Blischke and Murthy, 2000], and NORSOK Z-016 [NORGE], Stage Gate process [Crowe and Feinberg, 2001], and Reliability Enhancement Methodology and Modeling (REMM) methodology [Jones et al., 2003; REMM, 2006]. These are explained in Table 3.8.

Table 3.8 Reliability enhancement methods (Continue)

MIL-STD-785 [Blischke and Murthy, 2000]	This program originated in the United States for use in government and military acquisitions of high-tech products. The standard consist of the following four phases. • Phase 1: Concept. This phase deals with the identification and exploration of alternative solutions or solution concepts to satisfy a validated need as stated in the request for proposal. • Phase 2: Demonstration/Validation. This phase requires the manufacturer to define the procedures for demonstrating the reliability performance of the product and its validation in terms of meeting the reliability measures stated in the request for proposal. • Phase 3: Full-Scale Engineering Development. This phase involves formulation of the detailed engineering design and construction of a prototype. • Phase 4: Production. This is the final phase, during which products are produced and finally delivered to the buyer.
ISO 9000 series [ISO, 2007]	ISO 9000 is a family of standards for quality management systems. For a manufacturer, some of the requirements in ISO 9001 would include: • a set of procedures that cover all key processes in the business; • monitoring manufacturing processes to ensure they are producing quality product; • keeping proper records; • checking outgoing product for defects, with appropriate corrective action where necessary; and • regularly reviewing individual processes and the quality system itself for effectiveness.
BS5760 [O'Connor, 2007]	BS5760 is a "guidelines" standard that describes methods for reliability achievement. It involves five phases — definition, design and development, production, install and commission, and function.

Table 3. 8 Reliability enhancement methods

SAE M-100 [Blischke and Murthy, 2000]	SAE M-110 is a guideline and evolved under the auspices of the society of Automotive Engineers, Inc., with the aim of improving the reliability and quality of automotive products and manufacturing processes associated with the automotive industry. It consists of five phases as follows. • Phase 1: Concept. In this phase, the user specifies the reliability and maintainability requirements. • Phase 2: Development Design. In this phase, the user should verify the supplier's capabilities to undertake the actions specified in the reliability and maintainability requirement and monitor the supplier's progress through scheduled design reviews. • Phase 3: Build and Install. During the phase, process variables affecting reliability and maintainability should be identified and targeted for control during the manufacture, assembly, and installation of the equipment. • Phase 4: Operation/Support. This phase deals with user and supplier reliability and maintainability activities during the operation of the equipment and the support needed. • Phase 5: Conversion and Decommission. This phase involves either an upgrade or the scrapping of the system.
NORSOK Z-016 [NORGE]	NORSOK Z-016:Regularity management & reliability technology The purpose of NORSOK Z-16 is to establish requirements and guidelines for systematic and effective planning, execution and use of reliability technology to achieve cost-effective solutions. It is also objective of the standard to arrive at a common understanding with respect to use of reliability technology in the various life cycle phases.
Stage Gate Process [Crowe and Feinberg, 2001]	The stage gate method is a concurrent engineering process of DfR activities. This method consists of five phases called stage gates as follows. • Stage Gate 1: understanding customer requirements • Stage Gate 2: design assessment reliability testing • Stage Gate 3: design maturity testing • Stage Gate 4: production screening • Stage Gate 5: reliability monitoring
REMM [REMM, 2006]	The REMM project aims to develop a holistic model of all reliability activities and functions used in a product life cycle and so enhance design and manufacturing processes of complex avionic systems.

These reliability programs focus the prototype development and manufacturing stages of system lifecycle using accelerated testing methods. Although they are effective, they are expensive because any unsatisfactory results of reliability testing require redesign and retest cycles until the results are satisfactory. If the reliability enhancement efforts are made at the design stage, the chances of redesign and retest cycles may be reduced. Therefore, it may be a cost-effective reliability enhancement method.

As a result of this demand, Design-for-Reliability (DfR) arose. General 37 DfR activities carried out across 12 divisions are surveyed and listed in Table 3.9. This shows general activities and processes for DfR. However, specific-domain reliability knowledge and design recommendation rules are disregarded. Since system failures must be understood in the specific system context, specific-domain reliability knowledge representation is important for system reliability enhancement.

Besides representation, sharing reliability knowledge among various engineering groups is also important for system reliability enhancement because the more engineering group are involved, the more the exchange of the reliability knowledge will become a bottleneck [SORMAN, 2006].

Table 3.9 DfR survey checklist [Blischke and Murthy, 2000]

Management
1. Goal setting for division 2. Priority of quality and reliability improvement 3. Management attention and follow-up (ownership of goals)
Engineering
4. Documented hardware design cycle 5. Reliability goal setting by product or module 6. Priority of reliability-improvement goals 7. Ownership of reliability goals 8. Design for reliability (DfR) training 9. Preferred technology selection (standardization) 10. Component qualification testing 11. Original equipment manufacturer (OEM) selection and qualification testing 12. Physical failure analysis of testing failures 13. Failure and root-cause analysis 14. Statistically designed engineering experiments 15. Design and stress derating rules 16. Design review and design rule checking 17. Failure-rate estimation (prediction) 18. Thermal design and measurements 19. Worst-case analysis 20. Failure modes and effects analysis (FMEA) 21. Environmental (margin) testing 22. STRIFE (cyclical, multi-stress) testing 23. Design defect tracking (DDT) 24. Lessons-learned databases
Manufacturing
25. Design for manufacturability (DfM) 26. Priority of quality and reliability goals 27. Ownership of quality and reliability goals 28. Quality-training programs 29. Statistical process control (SPC/SQC) 30. Internal process audits 31. Supplier process audits 32. Incoming inspection 33. Component-level burn-in 34. Assembly-level burn-in 35. Product-level burn-in 36. Manufacturing defect tracking 37. Corrective action reports

3.6 RELIABILITY OF ELECTRONIC PACKAGING SYSTEM

Potential paradigm shift in electronic packaging is leading to integration of multiple system functions, such as high speed digital, high bandwidth optical, analog or RF as well sensing functions, into one high performance package or module system [Kim et al., 2007]. In general, such integration is limited by reliability problems.

System-level reliability for electronic packaging systems that include various potential failures of packaging components has been studied by [Bestory et al., 2007; Pucha et al., 2004;Rassaian et al., 2007; Solomalala et al., 2007]. Pucha et al. present reliability assessment of various packaging components and design change algorithms for system integration and miniaturization. Rassaian and Lee propose a generalized multi-domain Rayleigh-Ritz stress analysis method to obtain the stress and strain fields for a variety of packaging styles under cyclic thermal environments. Bestory et al. proposes a novel method to perform electron packaging reliability simulation taking into account manufacturing errors using the Mote-Carlo method. Solomalala et al. propose a platform, in which all physics are treated with dedicated simulation programs, coordinated by a custom-developed core section.

These bottom-up studies only consider wearout failures and use limited failures metrics like N50. Therefore, they cannot represent different aspects of system reliability like random failures. These studies also require computer-aided reliability tools to solve system reliability problems in a timely manner.

CHAPTER 4

PROBLEM STATEMENT AND RESEARCH OBJECTIVES

Based on the literature review presented in the previous chapter, this chapter presents an overview of gaps in the existing literature, summarizes these issues in a problem statement, and presents research objectives that are needed in order to address the research problem.

4.1 SUMMARY OF GAPS IN EXISTING RESEARCH

From the critical review of literature in the SDfR context, the following research gaps are identified and summarized.

- **Lack of a unified reliability analysis method for system design**: Existing reliability analysis methods focus on partial aspects of system design for reliability. They predominantly focus on assessment, not allocation. Therefore, a new unified method is needed, as well as an associated framework that will incorporate all relevant aspects of system-level reliability for allocation as well as assessment purposes.

- **Lack of reliability allocation algorithms for complex systems**: The existing reliability allocation algorithms are limited to mostly series systems and do not consider complex systems that include both series and parallel structures. Therefore, new target reliability allocation algorithms for complex systems are necessary.

- **Need for defining effective reliability metrics that cover both random failures and wearout failures**: Existing literature has focused on random failures or wearout failures individually. The existing literature does not present a combined approach to address both random and wearout failures. Thus, multiple reliability metrics and reliability prediction models are required for covering both random failures and wearout failures. Using such combined reliability metrics and prediction models, both current systems as well as future systems can be designed for reliability.

- **Lack of design recommendations for system-level reliability**: Most existing methods focus on reliability assessment in a limited sense. However, in addition to system-level reliability assessment, it is also necessary to make automated design recommendations based on expert knowledge so that the system level reliability satisfies the target reliability.

4.2 PROBLEM STATEMENT

The problems addressed by this thesis can be summarized as follows:

The design of reliable electronic packaging systems in a systematic and timely manner requires a consistent and unified method for allocating, predicting, and assessing reliability and for recommending design changes at the component and system level with the consideration of both random and wearout failures.

Such a method that incorporates all relevant aspects of system reliability knowledge shall be represented explicitly in a computationally effective form so that reliability knowledge can be processed automatically for collaborative system design without misinterpretation or error by humans.

4.3 THESIS OBJECTIVES

The primary goal of this research is to develop a knowledge model that facilitates designing reliable electronic packaging systems in a systematic and timely manner incorporating all relevant aspects of system reliability knowledge. This research goal leads to four objectives as follows:

Objective 1: To develop a unified knowledge model for system design for reliability that can address both reliability allocation and assessment.

A new unified knowledge model that incorporates all relevant aspects of system-level reliability knowledge is necessary in order to realize SDfR in a systematic and timely manner. This knowledge model requires a new reliability analysis structure that supports physical assembly structures (AS), logical structures (LS), failure mode structures (FMS), and failure interactions and dependencies (FID). This reliability analysis structure provides a backbone for reliability allocation and assessment.

Objective 2: To demonstrate that the developed unified knowledge model supports a representative target reliability allocation method for complex systems consisting of series and parallel subsystems.

Target reliability allocation methods are necessary to enable target reliability allocation for complex systems. Achieving this objective will show how the developed unified knowledge model can support plausible target reliability allocation methods. In this way the knowledge model can likely be updated and evolved to accommodate various target reliability allocation methods.

The target reliability allocation method shall include reliability weighting methods and allocation algorithms. These weighting methods and allocation algorithms shall support both series and parallel structures as well as both random and wearout failures.

Objective 3: To demonstrate that the developed unified knowledge model supports representative reliability metrics for random and wearout failures.

It is necessary to utilize effective reliability metrics in order to account for random and wearout failures in allocation, prediction, and assessment of reliability. The method shall support multiple representative random and wearout reliability metrics to increase the likelihood that it can support designing a variety of current systems as well as future systems.

Objective 4: To demonstrate that the developed unified knowledge model supports a representative method for recommending design changes.

Methods for recommending design changes are necessary to enhance the reliability of multi–level microelectronic systems in a systematic manner. Representative design change recommendation methods may start with subsystem design changes and end with top-level system design changes according to the structure that the knowledge model shall accommodate. The employed representative method shall use rules that capture both statistics-based and physics-based reliability knowledge in specific design domains. These rules for design changes shall be represented explicitly in the developed

knowledge model so that design recommendation activities are executed automatically based on the difference between allocated and assessed reliabilities.

Objective 5: To implement the developed knowledge model in a computing framework, and to demonstrate the applicability of the framework using specific test cases and prototype system-level reliability tools.

The developed knowledge model shall be implemented and demonstrated so that its validity and applicability are verified. As for implementation, an information framework and prototype system-level reliability tools that are based on the Objective 1 knowledge model shall be developed. As for demonstration, electronic system design test cases that exercise and test the main knowledge model perspectives shall be developed. Successful implementation and demonstration will exhibit the computational effectiveness of the knowledge model.

CHAPTER 5

RELIABILITY OBJECT MODEL (ROM): A KNOWLEDGE MODEL FOR SDFR

This chapter presents a knowledge model for SDfR, referred to as Reliability Object Model (ROM). ROM consists of five components: ROM Metrics, ROM Algorithms, ROM-Structure, ROM-Library, and ROM-Tree. These ROM components are explained in detail from Section 5.2 to Section 5.6 after ROM is introduced in Section 5.1.

5.1 OVERVIEW OF RELIABILITY OBJECT MODEL (ROM)

5.1.1 Introduction to Knowledge Representation

According to Booker and McNamara [Booker and McNamara, 2005], a knowledge representation is useful in efforts to capture the concepts of complex design methods such as SDfR. A knowledge representation captures all the categories within a domain as well as the relationships that define their behaviors and interactions, and it does so in a computationally efficient form. Formally specified knowledge representations are used in developing databases, query languages, and intelligent tools.

For knowledge representations, several formalisms are available including semantic data modeling, object-oriented data modeling, and logic-based modeling approaches [Mocko, 2006].

Semantic data models (SDM) have been developed as an information modeling method that can capture and express the structure of an application domain. One classic SDM is Entity-Relationship (ER) model [Chen, 1976]. This ER model consists of entities and relationships to represent an application domain. Because of its simplicity and powerfulness, the ER model is a widely accepted in relational database design.

Object-oriented data models (OODM) were initiated by the object-oriented programming method. Object-oriented data models consist of classes and relationships to model concepts in a domain. One distinct feature of object-oriented data models is that they can model both data structures and activities in a domain by class attributes and methods. They can also efficiently model complex concepts in detail using inheritance relationships among classes. Representative object-oriented data models are EXPRESS [Schenck and Wilson, 1994], UML [Harmon and Watson, 1997], and SysML [SysML, 2006].

Logic-based modeling approaches have been developed for knowledge-based reasoning. Logic-based models consist of data structure definition and logic or rule definition in a domain. Representative Logic-based models are Prolog [Covington et al., 1996], Frames [Giarratano and Riley, 1994], and DL [Baader, 2003; Grosse et al., 2005].

Such knowledge representation formalisms are compared and criticized by [Eastman and Fereshetian, 1994], [Peak et al., 2004], and [Mocko, 2006]. They conclude that the right knowledge representation should be chosen depending on modeling purposes and domains.

5.1.2 Object-Oriented Approach for Representing SDfR Knowledge

Among various knowledge representation formalisms, object-oriented data modeling approaches are most effective for representing both process and product knowledge of complex design concepts [Geymayr and Ebecken, 1995; Gorti et al., 1998; Peak et al., 2004; Patterson-Hine and Koen, 1989]. Therefore, we represent SDfR process and product knowledge using object-oriented data modeling approaches.

An object-oriented data model that forms a basis for knowledge representation is presented by Wong and Sriram [Wong and Sriram, 1993]. This is called SHARED Model. A SHARED object, $\mathbf{o_i}$, is described in the following form:

$$\mathbf{o_i} \equiv (\mathbf{oid}, \mathbf{A}, \mathbf{M}, \mathbf{R}) \tag{5.1}$$

Where

oid is a unique identifier of an object ($\mathbf{o_i}$). The set of all unique identifiers is denoted by **soid**.

A is a set of attributes. Each attribute is a set of three-tuples, (**aname**, **t**, **v**). The attribute has a unique identifier, **aname,** and it is described by a type, **t**, and a value, **v**. Each **t** has an associated domain, **domain(t)**, such that $\mathbf{v} \in (\mathbf{domain(t)} \cup \{\})$. In general, primitive types (domains) are defined such as real ($\mathbb{R}$), integer ($\mathbb{I}$), boolean ($\mathbb{B}$), character ($\mathbb{C}$), and string ($\mathbb{S}$).

M is a set of methods. Each method within the set is defined by (**mname**, **code**), where **mname** is a unique identifier for the method, and **code** is its description in an appropriate language.

R is a set of relationships among **o** and other objects. Each relationship is identified by its unique identifier, **rid**.

For example, a triangle geometric object is described as:

$\mathbf{o_1}$ = *(triangle_001,*
{(area, real, 12.0), (perimeter, real, 12.0)},
{(draw(),null), (fill(),null), (delete(),null)},
triangle_points_001)

A generic SHARED relationship that represents a relationship among objects is also described in the following form:

$$\mathbf{r_i} \equiv (\mathbf{rid}, \mathbf{rt}, \mathbf{RL}) \tag{5.2}$$

where

rid is a unique identifier of a relationship ($\mathbf{r_i}$). The set of all unique identifiers is denoted by **srid**.

rt is a relationship type.

RL is a set of relationship roles. Each role is a set of three-tuples, (**rlname**, **t**, **v**). **rlname** is the name of the role, **v** is the value of the role, and **t** is the type of **v** ($\mathbf{v} \in$ **domain(t),** where **domain(t)** is a subset of **soid**)

For example, the *triangle_points_001* relationship is described as:

$\mathbf{r_1}$ = *(triangle_points_001,*
has_part,
{(composite, triangle, triangle_001),
(components, point[3], [point_001, point_002, point_003])})

These objects and relationships can be understood more clearly in abstract-level. The abstract-level description of objects is called class, and the abstract-level description of relationships is called class relationship.

A class, $\mathbf{c_i}$, is described in the following form:

$$\mathbf{c_i} \equiv (\mathbf{cname}, \mathbf{AD}, \mathbf{MD}) \tag{5.3}$$

where

cname is a unique name of a class ($\mathbf{c_i}$). The set of all **cname** is denoted by **scname**.

AD is a set of attribute definitions. Each attribute definition is a set of two-tuples, (**pname**, **t**). The attribute definition has a unique identifier, **pname,** and it is described by a type, **t**

MD is a set of method definitions. Each method definition within the set is defined by (**mname**), where **mname** is a unique identifier for the method.

For example, a triangle geometric class is described as:

$\mathbf{c_1}$ = *(triangle,*
{(area, real), (perimeter, real))},
{(draw()),(fill()), (delete())})

A class relationship, $\mathbf{cr_i}$, is also described in the following form:

$$\mathbf{cr_i} \equiv (\mathbf{crid}, \mathbf{rt}, \mathbf{RLD}) \tag{5.4}$$

where

crid is a unique identifier of a class relationship ($\mathbf{cr_i}$).

rt is a relationship type.

RLD is a set of relationship role definitions. Each role definition is a set of two-tuples, (**rlname**, **t**). **rlname** is the name of the role definition, and **t** is a type (**domain(t)** is a subset of **scname)**.

For example, a relationship between the triangle class and the point class is described as:

$\mathbf{cr_1}$ = *(triangle_points,*
has_part,
{(composite, triangle),
(components, point[3])})

The object-level is also called instance-level, and the class-level is also called schema-level. Since an object is an instantiation of a class, a relationship between an object and a class is called instantiation. This relationship is expressed in mathematical form in Equations (5.5) and (5.6).

$$\mathbf{o_i} \in^{i} \mathbf{c_j} \tag{5.5}$$

$$\mathbf{r_m} \in^{i} \mathbf{cr_n} \tag{5.6}$$

where

$\mathbf{o_i}$ is an object that is instance of $\mathbf{c_j}$.

$\mathbf{c_j}$ is a class.

$\mathbf{r_m}$ is an object relationship that is instance of $\mathbf{cr_n}$.

$\mathbf{cr_n}$ is a class relationship.

$\in^{\mathbf{i}}$ represents an instantiation relationship.

5.1.3 Introduction to Reliability Objects

Wong and Sriram's SHARED model is adapted for representing reliability process and product knowledge, which is called Reliability Object. We define a Reliability Object as follows:

Definition 5-1

*A **Reliability Object (ro_i)** is an item that includes reliability metrics, reliability activities, or reliability structure information. It is used in reliability allocation, prediction, and assessment, and design recommendations.*

The definition is described in following mathematical form in Equation (5.7), which is based on the SHARED model.

$$\mathbf{ro_i} \equiv (\mathbf{roid}, \mathbf{RM}, \mathbf{RA}, \mathbf{RS}) \tag{5.7}$$

where

roid is a unique identifier of a reliability object ($\mathbf{ro_i}$). The set of all unique identifiers is denoted by **sroid**.

RM is a set of reliability metrics. Each metric is a set of three-tuples, (**rmname**, **t**, **v**). The metric has a unique identifier, **rmname,** and it is described by a type, **t**, and a value, **v**.

RA is a set of reliability activities. Each activity within the set is defined by (**raname**, **algorithm**), where **raname** is a unique identifier for the activity, and **algorithm** is its description in an appropriate language.

RS is a set of relationships among **ro** and other reliability objects that constructs reliability structure. Each relationship is identified by its unique identifier, **rsid**.

In the same way, we define a Reliability Relationship as follows:

Definition 5-2

*A **Reliability Relationship (rr_i)** is a relationship that represents a relation in terms of failures among reliability objects.*

The definition is described in following mathematical form in Equation (5.8), which is based on the SHARED model.

$$\mathbf{rr_i} \equiv (\mathbf{rrid}, \mathbf{rt}, \mathbf{RRL}) \tag{5.8}$$

where

rrid is a unique identifier of the reliability relationship (**rr_i**). The set of all unique identifiers is denoted by **srrid**.

rt is a relationship type.

RRL is a set of roles of reliability relationship. Each role is a set of three-tuples, (**rrlname**, **t**, **v**). **rrlname** is the name of the role, **v** is the value of the role, and **t** is the type of **v** (**v** $\in$ **domain(t),** where **domain(t)** is a subset of **sroid).**

For representing reliability objects and relationships in the schema-level, we use UML Class Diagram [Harmon and Watson, 1997] because a graphical representation is better for human understanding, and UML is a popular standard in industries [Peak et al., 2004]. The symbols of UML Class Diagram are illustrated and described in Figure 5.1. The class symbol of UML Class Diagram consists of three compartments: a class name, attributes, and operations. For example, a class for cars has a class name (i.e. *car*), two attributes (i.e. *speed* and *direction*), and two operations (i.e. *drive()* and *stop()*).

Classes populated in UML Class Diagram are connected each other by UML class relationships. Figure 5.1 introduces three representative UML class relationships:

generalization, composition, and association. Generalization, which is symbolized by a blank triangle and a line, is a relationship between a class and one or more refined versions of the class. For example, a sedan is a type of car, so the *car* class has a generalization relationship with the *sedan* class. Composition, which is symbolized by a black diamond and a line, is a relationship between a class and its part classes. For example, a car has tires as parts, so the *car* class has a composition relationship with the *tire* class. Association, which is symbolized by a line, is a relationship between a class and its associated classes. For example, cars are produced by a manufacturer, so the *car* class has an association relationship with the *manufacturer* class.

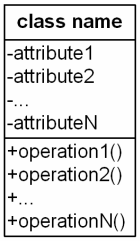
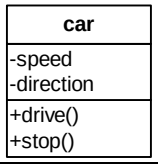
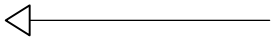
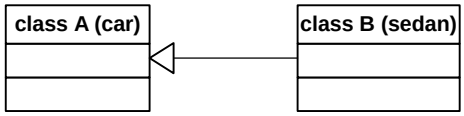
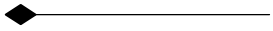
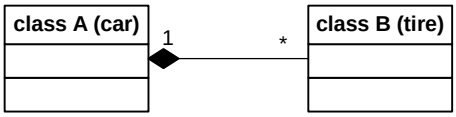
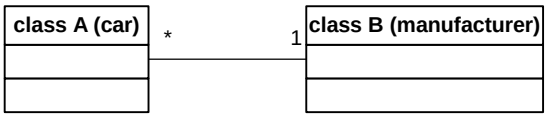
Symbols	Descriptions
1. Class	**$c_i \equiv$ (cname, AD, MD)**
	A class is an abstract level group of objects with similar properties (attributes), common behaviors (operations), common relationships to other objects, and common semantics.
	Example: a car has attributes and operations. 
2. Class Relationships	**$cr_i \equiv$ (crid, rt, RLD)**
rt_1 : generalization 	A generalization is a relationship between a class and one or more refined versions of the class. The class being refined is the superclass; each refined version is a subclass. Each subclass inherits all the features of its superclass. Through this relationship, classes are arranged into hierarchies.
	Example: A sedan is a type of car. 
rt_2 : composition 	A composition is a special, strong form of an aggregation. With composition relationship, the part objects are usually expected to live and die with the whole object.
	Example: A car has tires as parts. 
rt_3 : association 	An association is an abstract-level link, through which physical connections between objects are realized.
	Example: Cars are produced by a manufacturer. 

Figure 5.1 Symbols of UML Class Diagram

5.1.4 Introduction to Reliability Object Model (ROM)

While reliability objects represent a group of SDfR specifications (i.e. reliability metrics, reliability activities, reliability structure, and failure relationships), SHARED objects represent the other group of SDfR specifications (i.e. design features, reliability prediction models, and usage conditions). Therefore, whole representation of SDfR specifications is achieved by integrating reliability objects and SHARED objects. This is called Reliability Object Model (ROM). We define a ROM as follows:

Definition 5-3

*A **ROM (ROM$_i$)** is a set of reliability objects and SHARED objects. It represents reliability knowledge of a system design.*

The definition of ROM is described in mathematical form in Equation (5.9).

$$\mathbf{ROM_i} \equiv (\mathbf{RO} \cup \mathbf{RR})_{\text{for AS,FID,FMS,LS,RA, and RM}} \cup (\mathbf{O} \cup \mathbf{R})_{\text{for DF,RPM, and UC}} \quad (5.9)$$

where

RO is a set of reliability objects ($\mathbf{ro_i}$).

RR is a set of reliability relationships ($\mathbf{rr_i}$).

O is a set of objects ($\mathbf{o_i}$).

R is a set of relationships ($\mathbf{r_i}$).

The definition of relationship roles in $\mathbf{r_i}$ is extended, such that **domain(t)** is a subset of **soid** $\cup$ **sroid**.

We have developed ROM following three object-oriented modeling aspects: 1) defining attributes, 2) defining operations, 3) constructing a class structure. They

correspond to the three ROM aspects: 1) defining SDfR metrics, and 2) defining SDfR algorithms, and 3) constructing an SDfR structure.

The first aspect of ROM development is to define SDfR metrics. These are called ROM Metrics and described in Section 5.2. The second aspect of ROM development is to define SDfR algorithms. These are called ROM Algorithms and described in Section 5.3. The third aspect of ROM development is to construct an SDfR structure that supports seven SDfR specifications: assembly structures (AS), logical structures (LS), failure mode structures (FMS), failure interactions and dependencies (FID), design features (DF), reliability prediction models (RPM), and usage conditions (UC). AS, LS, FMS, and FID for reliability allocation and assessment belong to the generic system design domain. DF, RPM, and UC for reliability prediction and design change recommendation belong to specific system design domains, such as the printed wiring board assembly (PWBA) design domain or the mechanical power train assembly domain. Therefore, we separate the seven specifications into two representation groups: the representation of AS, LS, FMS, and FID, and the representation of DF, RPM, and UC. Since AS, LS, FMS, and FID are used for constructing a general reliability analysis structure, the representation is called ROM-Structure. This is defined in Equation 5.10 and described in Section 5.4. Since DF, RPM, and UC are reusable for each design in a given specific system design domain, the representation is called ROM-Library. This is defined in Equation 5.11 and described in Section 5.5.

$$\mathbf{ROM\text{-}Structure_i} \equiv (\mathbf{RO} \cup \mathbf{RR})_{\text{ for AS, LS, FMS, and FID}} \tag{5.10}$$

$$\mathbf{ROM\text{-}Library_i} \equiv (\mathbf{O} \cup \mathbf{R})_{\text{ for DF, RPM, and UC}} \tag{5.11}$$

For easier human understanding of the SDfR structure (ROM-Structure and ROM-Library), we have developed equivalent symbols and options to graphically represent this, which we call Reliability Object Model Tree (ROM-Tree).

The five ROM components — ROM Metrics, ROM Algorithms, ROM-Structure, ROM-Library, and ROM-Tree — are illustrated in Figure 5.2 and described in detail in the following sections.

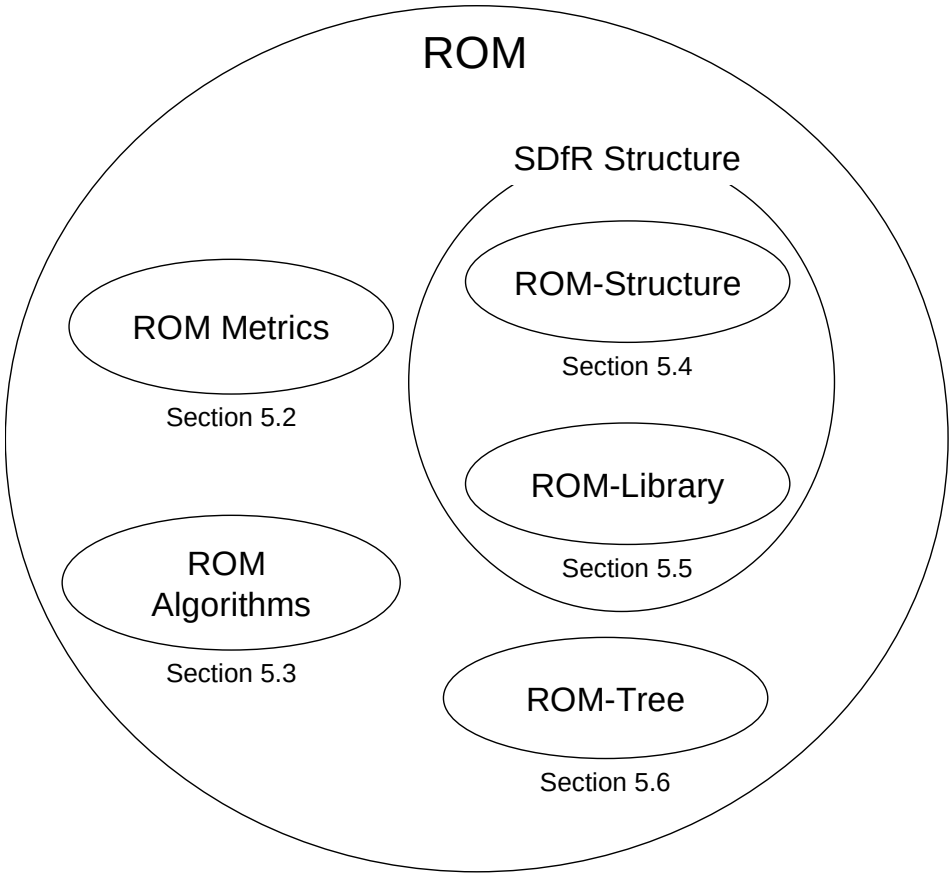
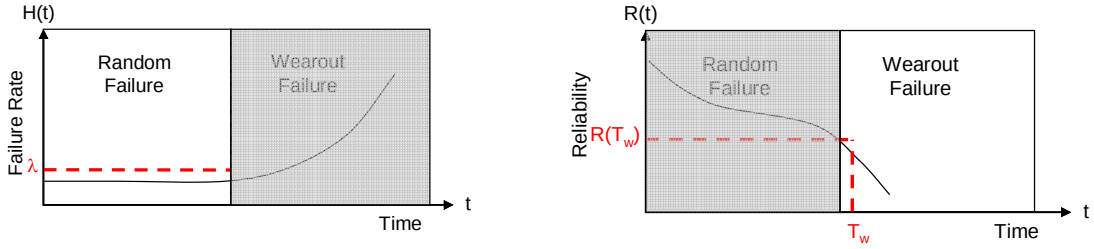


Figure 5.2 Overview of ROM

5.2 ROM METRICS



(a) Constant Random failure rate

(b) Percentile Wearout failure reliability

Figure 5.3 ROM Metrics[6]

To support various failures in systems [Condra, 1993; Foucher et al., 2002; Jensen, 1995], we define two basic reliability metrics (Figure 5.3): ***constant random failure rate*** ***(λ)*** and ***percentile wearout failure reliability ($R_w(T_w)$) at target-time-to-wearout-failure (T_w)***.

The constant random failure rate[7] represents reliability functions of the random failure stage (Equation (1.2)), and the percentile wearout failure reliability[8] is defined for measuring wearout failures at a given target time (Equation (1.3)).

We employ both metrics in ROM for reliability allocation, prediction, and assessment. Since the metrics have different purposes in each activity, they are represented differently. For example, the random failure rate for reliability allocation is Target Random Failure Rate (λ_{target}), and the random failure rate for reliability assessment

[6] We ignore the early failure stage of the bath-tub curve assuming systems or components with early failures are screened out and removed in the burn-in process.
[7] We call this as random failure rate afterward.
[8] We call this as wearout failure reliability afterward.

is Assessed Random Failure Rate ($\lambda_{assessed}$). All ROM Metrics for SDfR are represented in Table 5.1.

Table 5.1 Representation of ROM metrics

Metrics Names	Symbols	Representation
Reliability Allocation		
Target Random Failure Rate	λ_{target}	$\lambda_{i,target} = (\lambda_{target}, \text{real}, \text{null})$ $\lambda_{i,target} \in RM^*$
Target Time-to-Wearout-Failure	T_w	$T_{w,i} = (T_w, \text{real}, \text{null})$ $T_{w,i} \in RM$
Target Wearout Failure Reliability	$R_w(T_w)_{target}$	$R_w(T_{w,i})_{,i,target}$ $= (R_w(T_w)_{target}, \text{probability}, \text{null})$ $R_w(T_{w,i})_{,i,target} \in RM$
Reliability Prediction		
Predicted Random Failure Rate	$\lambda_{predicted}$	$\lambda_{i,predicted} = (\lambda_{predicted}, \text{real}, \text{null})$ $\lambda_{i,predicted} \in RM$
Predicted Wearout Failure Reliability	$R_w(T_w)_{predicted}$	$R_w(T_{w,i})_{,i,predicted}$ $=(R_w(T_w)_{predicted}, \text{probability}, \text{null})$ $R_w(T_{w,i})_{,i,predicted} \in RM$
Reliability Assessment		
Assessed Random Failure Rate	$\lambda_{assessed}$	$\lambda_{i,assessed} = (\lambda_{assessed}, \text{real}, \text{null})$ $\lambda_{i,assessed} \in RM$
Assessed Wearout Failure Reliability	$R_w(T_w)_{assessed}$	$R_w(T_{w,i})_{,i,assessed}$ $= (R_w(T_w)_{assessed}, \text{probability}, \text{null})$ $R_w(T_{w,i})_{,i,\ assessed} \in RM$
* RM is a set of reliability metrics defined in Equation (5.7). Each metric is a set of three-tuples, (rmname, t, v). The metric has a unique identifier, rmname, and it is described by a type, t, and a value, v.		

5.3 ROM ALGORITHMS

This section explains four types of reliability algorithms for SDfR: target reliability allocation algorithms, reliability prediction algorithms, reliability assessment algorithms, and design change recommendation rules. To help understanding such reliability algorithms, we make a list of reliability activities for SDfR in Section 5.3.1 and describe four basic reliability equations in Section 5.3.2. The overview of this section is illustrated in Figure 5.4.

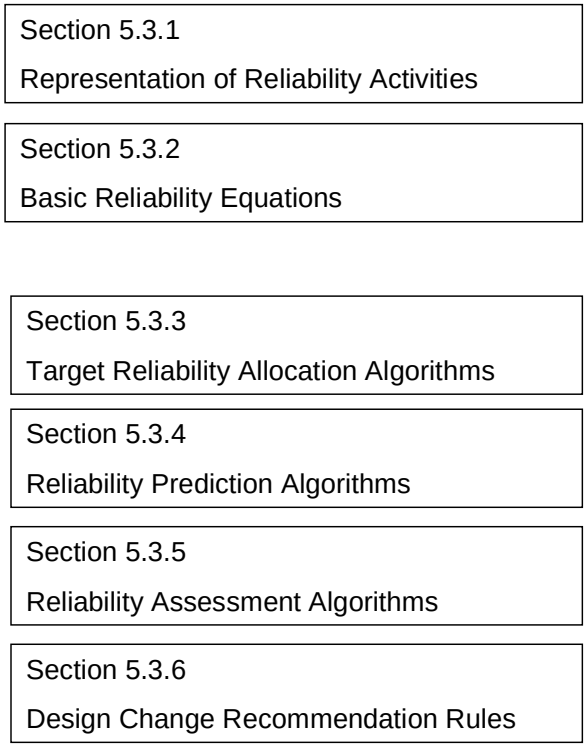


Figure 5.4 Overview of Section 5.3

5.3.1 Representation of Reliability Activities

Table 5.2 Representation of reliability activities

Activity Names	Symbols	Representation
Target Reliability Allocation	$RA_{allocate}$	$RA_{i,allocate} \in RA^{*}$
Target Random Failure Reliability Allocation - for series - for parallel	$RA_{allocate\_Rr}$ - $RA_{allocate\_Rr\_for\_series}$ - $RA_{allocate\_Rr\_for\_parallel}$	$RA_{i,allocate\_Rr} \in RA$ - $RA_{i,allocate\_Rr\_for\_series} \in RA$ - $RA_{i,allocate\_Rr\_for\_parallel} \in RA$
Target Wearout Failure Reliability Allocation - for series - for parallel	$RA_{allocate\_Rw}$ - $RA_{allocate\_Rw\_for\_series}$ - $RA_{allocate\_Rw\_for\_parallel}$	$RA_{i,allocate\_Rw} \in RA$ - $RA_{i,allocate\_Rw\_for\_series} \in RA$ - $RA_{i,allocate\_Rw\_for\_parallel} \in RA$
Reliability Prediction	$RA_{predict}$	$RA_{i,predict} \in RA$
Random Failure Reliability Prediction	$RA_{predict\_Rr}$	$RA_{i,predict\_Rr} \in RA$
Wearout Failure Reliability Prediction	$RA_{predict\_Rw}$	$RA_{i,predict\_Rw} \in RA$
Reliability Assessment	RA_{assess}	$RA_{i,assess} \in RA$
Random Failure Reliability Assessment - for series - for parallel	RA_{assess_Rr} - $RA_{assess\_Rr\_for\_series}$ - $RA_{assess\_Rr\_for\_parallel}$	$RA_{i,assess\_Rr} \in RA$ - $RA_{i,\ assess\_Rr\_for\_series} \in RA$ - $RA_{i,\ assess\_Rr\_for\_parallel} \in RA$
Wearout Failure Reliability Assessment - for series - for parallel - for independent FMs - for superimposable FMs	RA_{assess_Rw} - $RA_{assess\_Rw\_for\_series}$ - $RA_{assess\_Rw\_for\_parallel}$ - $RA_{assess\_Rw\_for\_ind}$ - $RA_{assess\_Rw\_for\_sup}$	$RA_{i,assess\_Rw} \in RA$ - $RA_{i,assess\_Rw\_for\_series} \in RA$ - $RA_{i,assess\_Rw\_for\_parallel} \in RA$ - $RA_{i,assess\_Rw\_for\_ind} \in RA$ - $RA_{i,assess\_Rw\_for\_sup} \in RA$
Design Recommendation	$RA_{recommend}$	$RA_{i,recommend} \in RA$
Design Recommendation for Random Failures	$RA_{recommend\_Rr}$	$RA_{i,recommend\_Rr} \in RA$
Design Recommendation for Wearout Failures	$RA_{recommend\_Rw}$	$RA_{i,recommend\_Rw} \in RA$

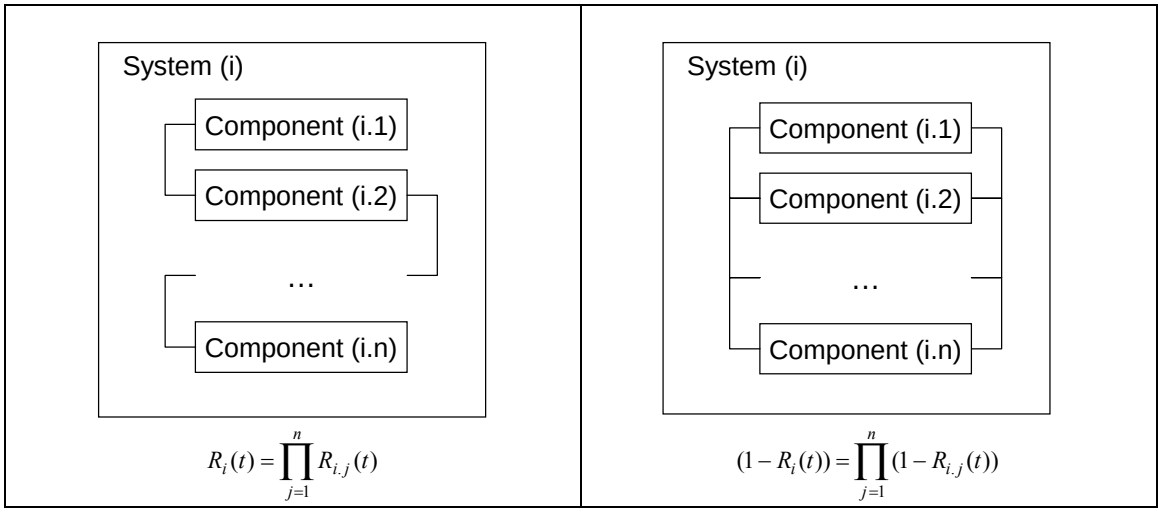
* RA is a set of reliability activities defined in Equation (5.7). Each activity within the set is defined by (raname, algorithm), where raname is a unique identifier for the activity, and algorithm is its description in an appropriate language.

We specify four reliability activities in Table 2.2 depending on failure types, logical structures, and failure mode structures. For example, $RA_{allocate\_Rr\_for\_series}$ represents a random failure reliability allocation activity for series structures, and $RA_{allocate\_Rw\_for\_series}$ represents a wearout failure reliability allocation activity for series structures. All reliability activities for SDfR are listed and represented in Table 5.2.

5.3.2 Basic Reliability Equations

We introduce basic reliability equations in this section: 1) a series structure equation, 2) a parallel structure equation, 3) an independent failure mode structure equation, and 4) a superimposable failure mode structure equation.

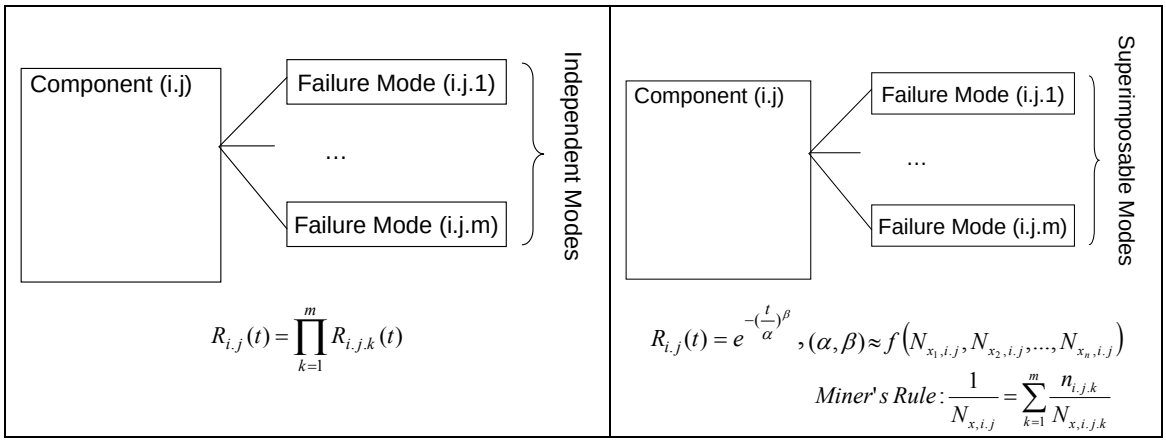
Systems are constructed by series and parallel structures to fulfill the required functions. A series structure indicates that a system fails if any item in the system fails, and a parallel structure indicates that a system works if any item in the system works. These two structures [Birolini, 2004] are illustrated in Figure 5.5.



(a) Series structure (b) Parallel structure

Figure 5.5 Logical structures

Similar to system failures based on logical structures, components fail by independent or superimposable failure mode structures. An independent failure mode structure means that a component fails due to any failure mode that is independent from other failure modes (Figure 5.6-(a)) [Leemis, 1995]. A superimposable failure mode structure means that a component fails due to the combination of multiple failure modes whose effects are superimposable (Figure 5.6-(b)). This superimposable failure mode structure is based on the Miner's Rule [Dowling, 1999]. This function is described in detail in Appendix B.



(a) Independent failure mode structure

(b) Superimposable failure mode structure

Figure 5.6 Failure mode structures

These four basic reliability equations are summarized in Table 5.3, which replace the 1-out-of-n parallel structure equation (Figure 5.5-(b)) by a generalized k-out-of-n parallel structure equation. In addition, two different types of parallel structure equations are described. One is a homogeneous redundancy structure equation, and the other is a heterogeneous redundancy structure equation.

Table 5.3 Basic reliability equations

Groups of systems or components			
	Series structure		$R_i(t) = \prod_{j=1}^{n} R_{i.j}(t)$ (5.12)
	Parallel structure		
		Homogeneous redundancy (k-out-of-n)	$R_i(t) = Homogeneous\ Redundancy(R_{i.j}(t), n, k)$ $R_i(t) = \sum_{l=k}^{n} \binom{n}{l} R_{i.j}^{l}(t)(1 - R_{i.j}(t))^{n-l}$ $(R_{i.1}(t) = R_{i.2}(t) = \ldots = R_{i.n}(t))$ (5.13)
		Heterogeneous redundancy (k-out-of-n)	$R_i(t) = Heterogeneous\ Redundancy(R_{i.j}(t), n, k)$ $R_i(t) = \sum_{l=k}^{n} \left[\sum^{c_l^n} \{ \prod^{l} R_{i.j,selected}(t) \prod^{n-l} (1 - R_{i.j,not\ selected}(t)) \} \right]$ (5.14)
Groups of failure modes			
	Independent failure modes structure		$R_{i.j}(t) = \prod_{k=1}^{m} R_{i.j.k}(t)$ (5.15)
	Superimposable failure modes structure		$R_{i.j}(t) = e^{-(\frac{t}{\alpha})^{\beta}}$, $(\alpha, \beta) \approx f\left(N_{x_1,i.j}, N_{x_2,i.j}, \ldots, N_{x_n,i.j}\right)$ (5.16) $Miner's\ Rule: \frac{1}{N_{x,i.j}} = \sum_{k=1}^{m} \frac{n_{i.j.k}}{N_{x,i.j.k}}$

i: item of interest (a system or a group),
i.j: sub-item i.j of item i (subsystems, subgroups, or components),
i.j.k: failure mode i.j.k of component i.j,
n: number of sub-items,
m: number of failure modes,
$N_{x,i.j}$: number of cycles at x percent failures of component i.j,
$N_{x,i.j.k}$: number of cycles at x percent failures of failure mode i.j.k,
$n_{i.j.k}$: number of cycles of failure mode i.j.k in a unit time.

5.3.3 Target Reliability Allocation Algorithms

In this section, we explain three methods to determine reliability weights and five reliability allocation algorithms for SDfR. Three methods are 1) historical data-based relative target reliability weight method, 2) design information-based relative target reliability weight method, and 3) uniform relative target reliability weight method. Five reliability allocation algorithms included in ROM are 1) random failure reliability allocation algorithm for series structures, 2) random failure reliability allocation algorithm for parallel structures, 3) wearout failure reliability allocation algorithm for series structures, and 4) wearout failure reliability allocation algorithm for parallel structures, and 5) general reliability algorithms for complex structures.

Target reliability allocation is to allocate given target system reliability to the subsystems. One problem of target reliability allocation studies is that, if there is no established objective function such as cost or performance, then it is difficult to define the the best allocation solution. Therefore, various heuristic or strategic methods have been suggested depending on different contexts, as presented in Chapter 3.

One reason of allocating target reliability is to improve reliability. Therefore, appropriate target reliability allocation methods could be determined according to reliability improvement technology, cost, time, system development limitations, and so on with estimated subsystem reliability information.

Suppose subsystem reliabilities are estimated, and there is no established objective function. One possible method is to allocate the required reliability improvement portion to the most unreliable subsystem (what might be called a “weakest link” method). Another possible method is to allocate required reliability improvement

portion to subsystems according to their relative reliability weights. Both methods can make sense depending on context and assumptions. If it is possible to improve subsystem reliability by a large amount with little extra cost, then the first approach is appropriate. However, if all subsystems are optimized for reliability in the previous development and small reliability improvements are expected for each subsystem, then allocating subsystem reliabilities according to their relative reliability weights is an appropriate approach. This second approach also reduces errors caused by overestimating or underestimating subsystem reliabilities.

Target reliability allocation may be better understood by deriving the relation between subsystem and system reliabilities. We derive this relation from the series structure equation in Equation (5.12). We take the natural logarithm of both sides of Equation (5.12) to get the relation between the exponents of both sides. This leads to Equation (5.17). Dividing both sides by the left hand side leads to Equation (5.18). The ratio of the logarithm of subsystem reliability to the logarithm of system reliability is defined as reliability weight, $w_{i,j}$, shown in Equation (5.19). Thus Equation (5.18) can be re-written as Equation (5.20). A comparatively large subsystem reliability weight indicates that the associated subsystem has low reliability compared to other subsystems in the parent system.

$$\ln(R_i(t)) = \sum_{j=1}^{n} \ln(R_{i.j}(t)) \tag{5.17}$$

$$1 = \sum_{j=1}^{n} \frac{\ln(R_{i.j}(t))}{\ln(R_i(t))} \tag{5.18}$$

$$w_{i.j} = \frac{\ln(R_{i.j}(t))}{\ln(R_i(t))} \tag{5.19}$$

$$1 = \sum_{j=1}^{n} w_{i.j} \tag{5.20}$$

where

i: systems,

i.j: subsystem j in system i ,

$w_{i.j}$: the reliability weight of subsystem j within system i.

Assuming that it the system has been previously optimized for reliability and that any desired increase in system reliability should therefore be allocated according to the previously obtained reliability weights, from Equation (5.19), target reliability of any subsystem ($R_{i.j,target}$) can be calculated using the given target reliability of the system ($R_{i,target}$) and an estimated target reliability weight of that subsystem ($w_{i.j}$). The estimated reliability weight is called Relative Target Reliability Weight (RTRW) in this research because the weight is only valid in a given system level for target reliability allocation.

In this section, three target reliability weighting methods are presented. These are ***historical data-based relative target reliability weight (RTRW^h)*** [Blischke and Murthy, 2000], ***design information-based relative target reliability weight (RTRW^d)***, and ***uniform relative target reliability weight (RTRW^u)*** [Blischke and Murthy, 2000]. The goal is to illustrate how the ROM approach can accommodate a variety of different allocation strategies. Which particular allocation strategy is most appropriate depends on the problem context and is not addressed in this thesis.

A first approach is based on historical data-based relative target reliability weights. When historical reliability data of subsystems are available and if it is decided that an

allocation according to this historical distribution remains appropriate, RTRW can be estimated easily. We derive $RTRW^h$ replacing the reliability in Equation (5.19) by the historical reliability data. This is mathematically expressed in Equation (5.21).

$$RTRW_{i.j}^{h} \equiv \frac{\ln(R_{i.j}^{h}(t))}{\ln(\prod_{j=1}^{n} R_{i.j}^{h}(t))} \tag{5.21}$$

where

$RTRW_{i.j}^{h}$: the historical data-based relative target reliability weight,

$R_{i.j}^{h}(t)$: the historical reliability data of the subsystem $_{i,j}$.

A second approach is based on design information-based relative target reliability weights. Historical reliability data are not always available. In this case relative reliability weights are estimated based on initial design information and expert experience. According to our electronic board reliability prediction experiences, we observe that the complexity of electronic boards is closely related to the reliability of the electronic boards. For example, the more components an electronic board includes, the less reliable the electronic board typically is assuming a serial reliability arrangement. In addition, the more complex components an electronic board includes, the less reliable the electronic board typically is. Therefore, the larger the complexity sum is, the less reliable the associated subsystem is.

The complexity of subsystems is also related to cost factors such as manufacturing cost, reliability improvement cost, and maintenance cost. In general, the more complex a subsystem is, the higher its manufacturing cost is expected to be. Therefore, an approach that allocates subsystem reliabilities according to their relative

complexity may be a reasonable approach for reducing unnecessary manufacturing cost caused by overestimating or underestimating subsystem reliabilities.

Based on such assumptions, we define $RTRW^d$ in mathematical form using variations of these two factors as shown in Equation (5.22). This complexity of components is defined to range from 0 to 10, with 10 being the largest complexity.

Determining the weights for two factors and determining complexity of components is based on domain expert experience. If reasonable weighting factors and guidelines are provided by domain experts, the $RTRW^d$ method is applicable in the design domain for allocating relative reliability weights.

$$\begin{aligned} RTRW_{i.j}^{d} &\equiv w_a \frac{ENC_{i.j}}{TENC_i} + w_b \frac{ECSCC_{i.j}}{TECSCC_i} \\ &\equiv w_a RENC_{i.j} + w_b RECSCC_{i.j} \end{aligned} \tag{5.22}$$

where

$RTRW_{i.j}^{d}$: design information-based relative target reliability weight of subsystem$_{i.j}$,

$ENC_{i.j}$: expected number of components in subsystem$_{i.j}$,

$TENC_i$: total expected number of components in system$_i$,

$ECSCC_{i.j}$: expected complexity sum of critical components in subsystem$_{i.j}$,

$TECSCC_i$: total expected complexity sum of critical components in system$_i$,

$RENC_{i.j}$: the ratio of $ENC_{i.j}$ to $TENC_i$,

$RECSCC_{i.j}$: the ratio of $ECSCC_{i.j}$ to $TECSCC_i$,

w_a: the weight of ENC factor for all $RTRW_{i.j}$,

w_b: the weight of ECSCC factor for all $RTRW_{i.j}$,

$w_a + w_b = 1$.

A final approach is based on uniform relative target reliability weights. When a system consists of n identical subsystems, a uniform target reliability weight can be assumed. In this case, target reliability is allocated based on the number of subsystems (n). This is mathematically expressed in Equation (5.23).

$$RTRW_{i.j}^{u} \equiv \frac{1}{n} \tag{5.23}$$

where

$RTRW_{i.j}^{u}$: the uniform relative target reliability weight of all subsystem j in system i ,

n: the total number of subsystems in system i .

With the above background on RTRW allocation methods and their assumptions, the following subsections explain five target reliability allocation algorithms that are based on these methods: 1) random failure reliability allocation for series structures (Section 5.3.3.1), 2) random failure reliability allocation for parallel structures (Section 5.3.3.2), 3) wearout failure reliability allocation for series structures (Section 5.3.3.3), 4) wearout failure reliability allocation for parallel structures (Section 5.3.3.4), and 5) reliability allocation for complex structures (Section 5.3.3.5).

5.3.3.1 Random failure reliability allocation for series structures

Random failure reliability is represented by an exponential function with a negative exponent, random failure rate (λ). If this random failure rate is applied to Equation (5.19), the relationship between the random failure rate of the system and the random failure rate of the subsystem is derived, shown in Equation (5.24). Out of this

equation, the random failure rate of the subsystem is estimated by the reliability weight of the subsystem and the random failure rate of the system. This is shown in Equation (5.25).

$$w_{i.j} = \frac{\ln(R_{i.j}(t))}{\ln(R_i(t))} = \frac{\ln(e^{-\lambda_{i.j}t})}{\ln(e^{-\lambda_i t})} = \frac{\lambda_{i.j}}{\lambda_i} \tag{5.24}$$

where

λ_i: the system target random failure rate,

$\lambda_{i.j}$: the subsystem target random failure rate,

$w_{i.j}$: the target reliability weight.

$$\lambda_{i.j} = \lambda_i \times w_{i.j} \tag{5.25}$$

where

λ_i: the system target random failure rate,

$\lambda_{i.j}$: the subsystem target random failure rate,

$w_{i.j}$: the target reliability weight.

This reliability allocation formula is demonstrated by a system that consists of four series subsystems. A given target random failure rate of the system is $\lambda_i = 1000$, and it is allocated based on the given RTRWs in Table 5.4. The results are also shown in Table 5.4.

For verifying the results, the random failure rate of the system is reversely calculated from the allocated random failure rates of the subsystems. The reverse calculation result is exactly the same as the given target random failure rate of the system (Table 5.4).

Table 5.4 Target random failure reliability allocation for series structures

System	Subsystem	$RTRW_{i.j}$	$\lambda_{i.j}$	Reverse Calculation $\lambda_{i.j}$
i		1.0	1000	1000
	i.1	0.1	100	100
	i.2	0.2	200	200
	i.3	0.3	300	300
	i.4	0.4	400	400

5.3.3.2 Random failure reliability allocation for parallel structures

While the random failure reliability allocation for series structures is straightforward, random failure reliability allocation for parallel structures is not, as the basic reliability equations of parallel structures in Table 5.3 are not straightforward. Therefore, a new numerical method for allocating random failure reliability for parallel structures is developed (Figure 5.7). This method consists of six steps, the details of which follow:

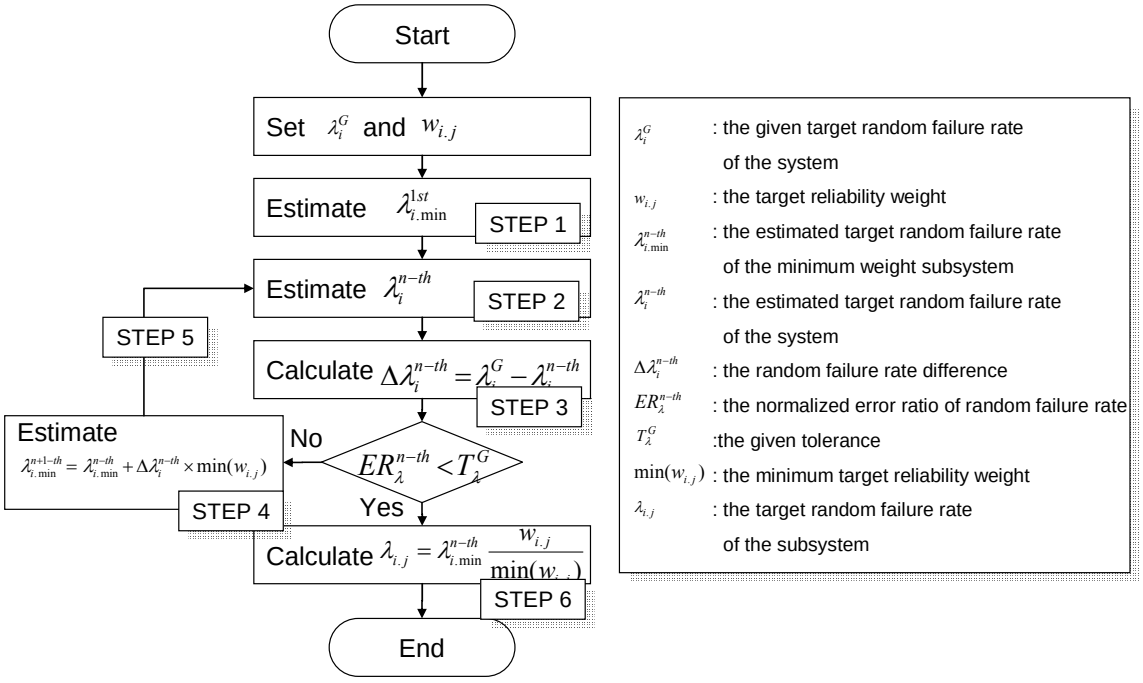


Figure 5.7 Flow chart of the random failure reliability allocation algorithm for parallel structures

STEP 1: Estimate the initial random failure rate of the minimum-weight subsystem ($\lambda_{i.\min}^{1st}$).

This method starts with guessing the initial random failure rate of the subsystem whose weight is the least among the subsystems. This is derived from the reliability equation for series structures, shown in Equation (5.26). From Equation (5.26), the reliability function of the system is expressed by the random failure rate of the system, and the exponent term (1/n) is replaced by the minimum-weight. Then, the reliability of the subsystem whose weight is minimum ($R_{r,i.\min}(t)$) is guessed (Equation (5.27)), and the random failure rate of the minimum-weight subsystem is calculated through the least squares method in a given time period ($0 \le t \le T_w$), shown in Equation (5.28).

$$R_{i.j}(t) = (R_i(t))^{1/n} \tag{5.26}$$

where

$R_i(t)$: the system reliability,

$R_{i.j}(t)$: the subsystem reliability,

n: the number of subsystems.

$$\begin{aligned} R_{r,i.\min}(t) &= (R_{r,i}(t))^{\min(w_{i.j})} \\ &= (e^{-\lambda_i^G t})^{\min(w_{i.j})} \end{aligned} \tag{5.27}$$

where

$R_{r,i.\min}(t)$: the random failure reliability of the minimum-weight subsystem,

$R_{r,i}(t)$: the random failure reliability of the system,

$\min(w_{i.j})$: the minimum-weight,

λ_i^G : the given target random failure rate of the system.

$$e^{-\lambda_{i.\min}^{1st} t} \approx R_{r,i.\min}(t) \tag{5.28}$$

where

$R_{r,i.min}(t)$: the random failure reliability of the minimum-weight subsystem,

$\lambda_{i.\min}^{1st}$: the initial random failure rate of the minimum-weight subsystem.

STEP 2: Estimate the random failure rate of the system (λ_i^{1st}).

From the initial random failure rate of the minimum-weight subsystem ($\lambda_{i.\min}^{1st}$), the random failure rates of other subsystems are calculated, shown in Equation (5.29).

$$\lambda_{i.j}^{1st} = \lambda_{i.\min}^{1st} \frac{w_{i.j}}{\min(w_{i.j})} \tag{5.29}$$

where

$\lambda_{i.j}^{1st}$: the initial random failure rate of the subsystem,

$\lambda_{i.\min}^{1st}$: the initial random failure rate of the minimum-weight subsystem,

$w_{i.j}$: the target reliability weight,

$\min(w_{i.j})$: the minimum target reliability weight.

Since all the random failure rates of the subsystems are guessed from Equation (5.29), the random failure reliability ($R_{r,i}^{1st}(t)$) of the system is estimated with the basic parallel structure equation. The random failure rate (λ_i^{1st}) of the system is calculated through the least squares method in a given time period ($0 \leq t \leq T_w$). This calculation for homogeneous redundancy systems is shown in Equation (5.30) and the calculation for heterogeneous redundancy systems is shown in Equation (5.31).

$$e^{-\lambda_i^{1st}t} \approx Homogeneous\ \ Redundancy\ (R_{r,i,j}^{1st}(t), n, k) \tag{5.30}$$

where

λ_i^{1st} : the estimated random failure rate of the system,

$R_{r,i,j}(t)$: the random failure reliability of the subsystem.

$$e^{-\lambda_i^{1st}t} \approx Heterogeneous\ Redundancy\ (R_{r,i,j}^{1st}(t), n, k) \tag{5.31}$$

where

λ_i^{1st} : the estimated random failure rate of the system,

$R_{r,i,j}(t)$: the random failure reliability of the subsystem.

STEP 3: Calculate the difference between the given target random failure rate of the system (λ_i^G) and the estimated random failure rate (λ_i^{1st}).

According to the difference between the given target random failure rate of the system (λ_i^G) and the estimated random failure rate of the system (λ_i^{1st}), the estimated random failure rate of the minimum-weight subsystem ($\lambda_{i.\min}^{1st}$) is evaluated. Equation (5.32) shows random failure rate difference ($\Delta\lambda_i^{1st}$) and Equation (5.33) shows normalized error ratio of random failure rate (ER_λ^{1st}).

$$\Delta\lambda_i^{1st} = \lambda_i^G - \lambda_i^{1st} \tag{5.32}$$

where

$\Delta\lambda_i^{1st}$: the random failure rate difference of the system,

λ_i^G : the given random failure rate of the system,

λ_i^{1st} : the estimated random failure rate of the system.

$$ER_{\lambda}^{1st} = \left| \frac{\lambda_i^G - \lambda_i^{1st}}{\lambda_i^G} \right| \tag{5.33}$$

where

ER_{λ}^{1st} : the normalized error ratio of random failure rate,

λ_i^G : the given random failure rate of the system,

λ_i^{1st} : the estimated random failure rate of the system.

STEP 4: Estimate the second time-guess random failure rate of the minimum-weight subsystem ($\lambda_{i.\min}^{2nd}$).

If the error ratio is larger than the given tolerance ($ER_{\lambda}^{1st} > T_{\lambda}^{G}$), the estimation of the next random failure rate of the minimum-weight subsystem ($\lambda_{i.\min}^{2nd}$) out of the previous subsystem random failure rate ($\lambda_{i.\min}^{1st}$) is necessary. An issue is how to find the relationship between the two random failure rates of the minimum-weight subsystem. Since the random failure rate difference of the system ($\Delta\lambda_i^{1st}$) is proportionally related to the expected random failure rate difference of the minimum-weight subsystem ($\Delta\lambda_{i.\min}^{expected} = \lambda_{i.\min}^{2nd} - \lambda_{i.\min}^{1st}$), the random failure rate difference of the system multiplied by the minimum-weight ($\Delta\lambda_i^{1st} \times \min(w_{i.j})$) is used for guessing the next random failure rate of the minimum-weight subsystem ($\lambda_{i.\min}^{2nd}$). Since the random failure rate difference of the system multiplied by the minimum-weight ($\Delta\lambda_i^{1st} \times \min(w_{i.j})$) is smaller than the expected random failure rate difference of the minimum-weight subsystem ($\Delta\lambda_{i.\min}^{expected}$), the estimation may be conservative but it is good for stable convergence. The final relationship is shown in Equation (5.34).

$$\lambda_{i.j}^{2nd} = \lambda_{i.j}^{1st} + \Delta\lambda_{i}^{1st} \times \min(w_{i.j}) \tag{5.34}$$

where

$\lambda_{i.\min}^{2nd}$: the second time-guess random failure rate of the minimum-weight subsystem,

$\lambda_{i.\min}^{1st}$: the initial random failure rate of the minimum-weight subsystem,

$\Delta\lambda_{i}^{1st}$: the random failure rate difference of the system.

STEP 5: Repeat STEPS 2, 3, and 4 until updated error ratio is smaller than a given tolerance ($ER_{\lambda}^{n-th} < T_{\lambda}^{G}$).

STEPS 2, 3, and 4 are repeated until updated error ratio is smaller than a given tolerance ($ER_{\lambda}^{n-th} < T_{\lambda}^{G}$).

STEP 6: Calculate the allocated target random failure rates of the subsystems ($\lambda_{i.j}^{n-th}$).

If the updated error ratio is smaller than the given tolerance ($ER_{\lambda}^{n-th} < T_{\lambda}^{G}$), then find the allocated target random failure rates of the subsystems based on the relative value between the weight of the subsystem and the minimum-weight. This relationship is shown in Equation (5.35).

$$\lambda_{i.j}^{n-th} = \lambda_{i.\min}^{n-th} \times \frac{w_{i.j}}{\min(w_{i.j})} \tag{5.35}$$

where

$\lambda_{i.j}^{n-th}$: the final random failure rate of the subsystem,

$\lambda_{i.\min}^{n-th}$: the final random failure rate of the minimum-weight subsystem,

$w_{i.j}$: the weight of the subsystem,

$\min(w_{i.j})$: the minimum-weight.

This algorithm is demonstrated by a system that consists of four parallel homogeneous subsystems. A given target random failure rate of the system $\lambda_i = 1000$ is allocated based on the given RTRWs in Table 5.5 and the algorithm described at the above (Equation (5.26) - Equation (5.35)) during the given time segment from 0 hour to 10000 hours. The results are also shown in Table 5.5.

For verifying the results, the system random failure rate is reversely calculated from the allocated subsystem random failure rates. The reverse calculation result is almost same as that of the given target random failure rate of the system with a 0.09% error due to numerical errors (Table 5.5).

Table 5.5 Target random failure reliability allocation for homogeneous parallel structures (1 out of 4)

System	Subsystem	$RTRW_{i.j}$	$\lambda_{i.j}$	Reverse Calculation $\lambda_{i.j}$
i		1	1000	1000.9
	i.1	0.25	45213	45213
	i.2	0.25	45213	45213
	i.3	0.25	45213	45213
	i.4	0.25	45213	45213

Another example of the reliability allocation algorithm is a system that consists of four parallel heterogeneous subsystems with the different target reliability weights. A given system target random failure rate $\lambda_i = 1000$ is allocated based on the given RTRWs in Table 5.6 and the algorithm described in Equations (5.26) - (5.35) during the given time segment from 0 hour to 10000 hours. The results are also shown in Table 5.6.

For verification of the results, the system random failure rate is reversely calculated from the allocated subsystem random failure rates. The reverse calculation

result is almost same as that of the given target random failure rate of the system with a 0.08% error due to numerical errors caused by the least squares method (see Table 5.6).

Table 5.6 Target random failure reliability allocation for heterogeneous parallel structures (1 out of 4)

System	Subsystem	$RTRW_{i.j}$	$\lambda_{i.j}$	Reverse Calculation $\lambda_{i.j}$
i		1.0	1000	1000.8
	i.1	0.1	20980	20980
	i.2	0.2	41960	41960
	i.3	0.3	62940	62940
	i.4	0.4	83920	83920

5.3.3.3 Wearout failure reliability allocation for series structures

Percentile wearout failure reliability is represented by the Weibull function at a given target-time-to-wearout-failure ($R_{w,i}(T_w)$). If this wearout failure reliability is applied to Equation (5.19), the relationship between the wearout failure reliability of a system and the wearout failure reliability of the subsystem is derived, shown in Equation (5.36). If the target wearout failure reliability of the system and the reliability weight of its subsystems are given, the target wearout failure reliability of its subsystems are calculated through Equation (5.37)

$$w_{i.j} = \frac{\ln(R_{w,i.j}(T_w))}{\ln(R_{w,i}(T_w))} \tag{5.36}$$

$$R_{w,i.j}(T_w) = R_{w,i}(T_w)^{w_{i.j}} \tag{5.37}$$

where

$R_{w,i}(T_w)$: the wearout failure reliability of the system,

$R_{w,i.j}(T_w)$: the wearout failure reliability the subsystem,

$w_{i.j}$: the target reliability weight.

This reliability allocation formula is demonstrated by a system that consists of four series subsystems. A given system target wearout failure reliability, $R_{w,i}(T_w) = 0.5$, is allocated based on the given RTRWs in Table 5.7. The results are also shown in Table 5.7.

For verification of the results, the system wearout failure reliability is reversely calculated from the allocated subsystem wearout failure reliability. The reverse calculation result is exactly same as the given target wearout failure reliability of the system (see Table 5.7).

Table 5.7 Target wearout failure reliability allocation for series structures

System	Subsystem	$RTRW_{i.j}$	$R_{w,i.j}(T_w)$	Reverse Calculation $R_{w,i.j}(T_w)$
i		1.0	0.5	0.5000
	i.1	0.1	0.9330	0.9330
	i.2	0.2	0.8706	0.8706
	i.3	0.3	0.8123	0.8123
	i.4	0.4	0.7579	0.7579

5.3.3.4 Wearout failure reliability allocation for parallel structures

Similar to target random failure reliability allocation for parallel structures, wearout failure reliability allocation for parallel structures is not straightforward. Therefore, a new numerical method for allocating wearout failure reliability for parallel structures is developed (see Figure 5.8). This method consists of six steps, the details of which follow:

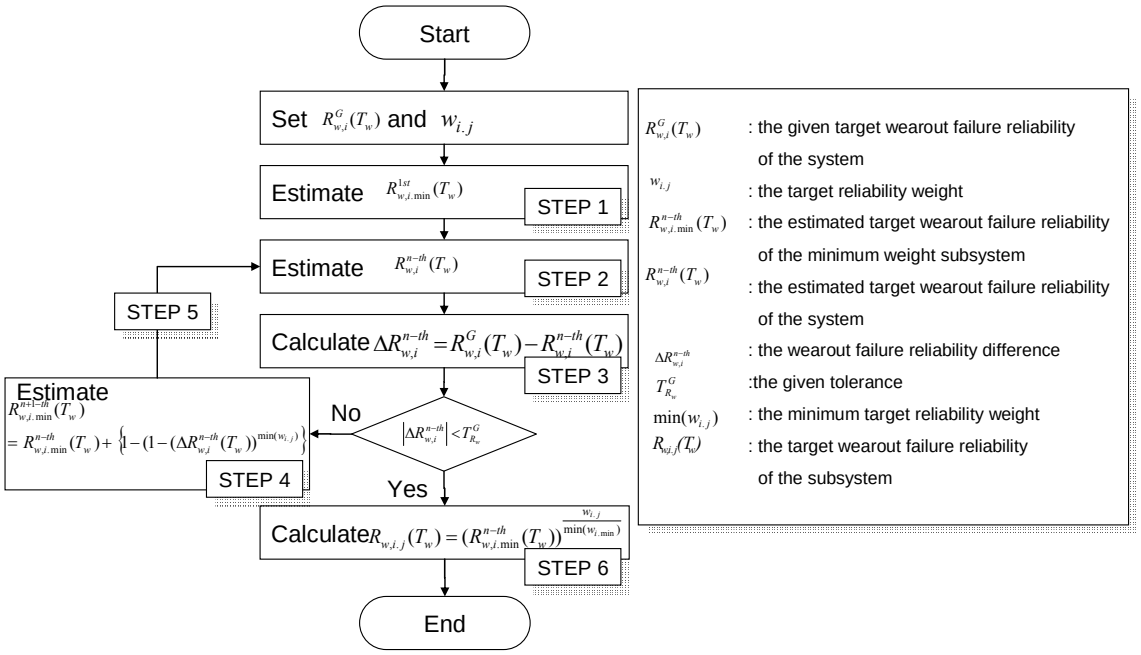


Figure 5.8 Flow chart of the wearout failure reliability allocation algorithm for parallel structures

STEP 1: Estimate the initial wearout failure reliability of the minimum-weight subsystem ($R^{1st}_{w,i.\min}(T_w)$).

This method starts with guessing the initial wearout failure reliability of the subsystem whose weight is the least among the subsystems through the simplified reliability equation of the series structure, shown in Equation (5.38).

$$R_{i.j}(t) = (R_i(t))^{1/n} \tag{5.38}$$

where

$R_i(t)$: the system reliability,

$R_{i.j}(t)$: the subsystem reliability,

n: the number of subsystems.

From Equation (5.38), the wearout failure reliability of the minimum-weight subsystem is estimated by replacing the reliability function and the exponent term (1/n) by percentile wearout failure reliability and the minimum-weight, shown in Equation (5.39).

$$R^{1st}_{w,i.\min}(T_w) = (R^{G}_{w,i}(T_w))^{\min(w_{i.j})} \tag{5.39}$$

where

$R^{1st}_{w,i.\min}(T_w)$: the wearout failure reliability of the minimum-weight subsystem,

$R^{G}_{w,i}(T_w)$: the given target wearout failure reliability of the system,

$\min(w_{i.j})$: the minimum-weight.

STEP 2: Estimate the wearout failure reliability of the system ($R_{w,i}^{1st}(T_w)$).

From the initial wearout failure reliability of the minimum-weight subsystem ($R_{w,i}^{1st}(T_w)$), other subsystem wearout failure reliabilities are calculated, shown in Equation (5.40).

$$R_{w,i.j}^{1st}(T_w) = R_{w,i.\min}^{1st}(T_w)^{\frac{w_{i.j}}{\min(w_{i.j})}} \tag{5.40}$$

where

$R_{w,i.j}^{1st}(T_w)$: the wearout failure reliability of the subsystem,

$R_{w,i.\min}^{1st}(T_w)$: the wearout failure reliability of the minimum-weight subsystem,

$w_{i.j}$: the target reliability allocation weight,

$\min(w_{i.j})$: the minimum target reliability weight.

Since the wearout failure reliability of all the subsystems is guessed from Equation (5.40), the wearout failure reliability ($R_{w,i}^{1st}(T_w)$) of the system is estimated with the basic parallel structure equation. The calculation for homogeneous redundancy systems is shown in Equation (5.41) and the calculation for heterogeneous redundancy systems is shown in Equation (5.42).

$$R_{w,i}^{1st}(T_w) = Homogeneous\ Redundancy\ \ (R_{w,i.j}^{1st}(T_w), n, k) \tag{5.41}$$

where

$R_{w,i}^{1st}(T_w)$: the wearout failure reliability of the system,

$R_{w,i.j}^{1st}(T_w)$: the wearout failure reliability of the subsystem,

$w_{i.j}$: the target reliability weight.

$$R_{w,i}^{1st}(T_w) = \textit{Heterogeneous Redundancy } (R_{w,i,j}^{1st}(T_w), n, k) \quad (5.42)$$

where

$R_{w,i}^{1st}(T_w)$: the wearout failure reliability of the system,

$R_{w,i,j}^{1st}(T_w)$: the wearout failure reliability of the subsystem,

$w_{i,j}$: the target reliability weight.

STEP 3: Calculate the difference between the given target wearout failure reliability of the system ($R_{w,i}^{G}(T_w)$) and the estimated wearout failure reliability of the system ($R_{w,i}^{1st}(T_w)$).

According to the difference between the given target wearout failure reliability of the system ($R_{w,i}^{G}(T_w)$) and the estimated wearout failure reliability of the system ($R_{w,i}^{1st}(T_w)$), the estimated wearout failure reliability of the minimum-weight subsystem ($R_{w,i.\min}^{1st}(T_w)$) is evaluated. Equation (5.43) shows wearout failure reliability difference ($\Delta R_{w,i}^{1st}$).

$$\Delta R_{w,i}^{1st} = R_{w,i}^{G}(T_w) - R_{w,i}^{1st}(T_w) \quad (5.43)$$

where

$\Delta R_{w,i}^{1st}(T_w)$: the wearout failure reliability difference of the system,

$R_{w,i}^{G}(T_w)$: the given wearout failure reliability of the system,

$R_{w,i}^{1st}(T_w)$: the estimated wearout failure reliability of the system.

STEP 4: Estimate the second time-guess wearout failure reliability of the minimum-weight subsystem ($R^{2nd}_{w,i.\min}(T_w)$).

If the magnitude of the wearout failure reliability difference is larger than a given tolerance ($\left|\Delta R^{1st}_{w,i}\right| > T^{G}_{R_w}$), the estimation of the next wearout failure reliability of the minimum-weight subsystem ($R^{2nd}_{w,i.\min}(T_w)$) out of the previous subsystem wearout failure reliability ($R^{1st}_{w,i.\min}(T_w)$) is necessary. A problem is finding the relationship between the two wearout failure reliability of the minimum-weight subsystem. The wearout failure reliability difference of the system ($\Delta R^{1st}_{w,i}(T_w)$) is related to the expected wearout failure reliability difference of the subsystem ($\Delta R^{expected}_{w,i.\min}(T_w) = R^{2nd}_{w,i.\min}(T_w) - R^{1st}_{w,i.\min}(T_w)$). We use the wearout failure reliability difference applied to 1-out-of-n parallel equation ($\left\{1-(1-(\Delta R^{n-th}_{w,i}(T_w))^{\min(w_{i.j})}\right\}$) for estimating the second time-guess wearout failure reliability of the minimum-weight subsystem ($R^{2nd}_{w,i.\min}(T_w)$). Since the result of $\left\{1-(1-(\Delta R^{n-th}_{w,i}(T_w))^{\min(w_{i.j})}\right\}$ is smaller than the expected wearout failure reliability difference of the minimum-weight subsystem ($\Delta R^{expected}_{w,i.\min}(T_w)$), the estimation may be conservative but it is good for stable convergence. The final relationship is shown in Equation (5.44).

$$R_{w,i.\min}^{2nd}(T_w) = R_{w,i.\min}^{1st}(T_w) + \left\{1-(1-(\Delta R_{w,i}^{n-th}(T_w))^{\min(w_{i.j})}\right\} \quad (5.44)$$

where

$R_{w,i.\min}^{2nd}(T_w)$: the second time-guess wearout failure reliability of the minimum-weight subsystem,

$R_{w,i.\min}^{1st}(T_w)$: the initial wearout failure reliability of the minimum-weight subsystem,

$\Delta R_{w,i}^{1st}(T_w)$: the wearout failure reliability difference of the system.

STEP 5: Repeat STEPS 2, 3, and 4 until updated difference is smaller than a given tolerance ($\left|\Delta R_{w,i}^{n-th}\right| < T_{R_w}^G$).

STEPS 2, 3, and 4 are repeated until updated difference is smaller than a given tolerance ($\left|\Delta R_{w,i}^{n-th}\right| < T_{R_w}^G$).

STEP 6: Calculate the wearout failure reliability of the subsystems.

If the updated difference is smaller than the given tolerance ($\left|\Delta R_{w,i}^{n-th}\right| < T_{R_w}^G$), then find the allocated target wearout failure reliability of the subsystems based on the relative value between the weight of the subsystem and the minimum-weight. This relation is shown in Equation (5.45).

$$R_{w,i.j}^{n-th}(T_w) = (R_{w,i.\min}^{n-th}(T_w))^{\frac{w_{i.j}}{\min(w_{i.j})}} \quad (5.45)$$

where

$R_{w,i.j}^{n-th}(T_w)$: the final wearout failure reliability of the subsystem,

$R_{w,i.\min}^{n-th}(T_w)$: the final wearout failure reliability of the minimum-weight subsystem,

$w_{i.j}$: the weight of the subsystem,

$\min(w_{i.j})$: the minimum-weight.

This algorithm is demonstrated by a system that consists of four parallel homogeneous subsystems. A given system target wearout failure reliability, $R_{w,i}(T_w)$ = 0.5, is allocated based on the given RTRWs in Table 5.8 and the algorithm described in Equations (5.38) - (5.45). The results are also shown in Table 5.8.

For verification of the results, the system wearout failure reliability is reversely calculated from the allocated subsystem wearout failure reliability. The reverse calculation result is almost same as the given target wearout failure reliability of the system with a 0.00% error (see Table 5.8).

Table 5.8 Target wearout failure reliability allocation for homogeneous parallel structures (1 out of 4)

System	Subsystem	$RTRW_{i.j}$	$R_{w,i.j}(T_w)$	Reverse Calculation $R_{w,i.j}(T_w)$
i		1.0	0.5	0.5000
	i.1	0.25	0.1591	0.1591
	i.2	0.25	0.1591	0.1591
	i.3	0.25	0.1591	0.1591
	i.4	0.25	0.1591	0.1591

Another example of the reliability allocation algorithm is a system that consists of four parallel heterogeneous subsystems with different RTRWs. A given system target wearout failure reliability, $R_{w,i}(T_w)$ = 0.5, is allocated based on the given RTRWs in Table 5.9 and the algorithm described in Equations (5.38) - (5.45). The results are also shown in Table 5.9.

For verification of the results, the system wearout failure reliability is reversely calculated from the allocated subsystem wearout failure reliability. The reverse

calculation result is almost same as the given target wearout failure reliability of the system with a 0.1% error owing to the numerical errors (see Table 5.9).

Table 5.9 Target wearout failure reliability allocation for heterogeneous parallel structures (1 out of 4)

System	Subsystem	$RTRW_{i.j}$	$R_{w,i.j}(T_w)$	Reverse Calculation $R_{w,i.j}(T_w)$
i		1.0	0.5000	0.5005
	i.1	0.1	0.3745	0.3745
	i.2	0.2	0.1403	0.1403
	i.3	0.3	0.0525	0.0525
	i.4	0.4	0.0197	0.0197

5.3.3.5 Reliability allocation for complex structures

For reliability allocation of complex structures that include multiple series and parallel structures, all reliability allocation algorithms described in the previous sections must be combined in a systematic manner. This recursive allocation algorithm is described in pseudo-code form in Table 5.10.

Before the algorithm starts, the target reliability of the top level system must be assigned. The first step is to set RTRWs among subsystems in the same subsystem level. The second step is to calculate the target reliability of the subsystems from the target reliability of the system and the subsystem RTRWs. The allocated subsystem target reliability values are subsequently used to determine the target reliability of its subsystems by recursively repeating the first and second steps until simple homogenous subsystems are reached.

Table 5.10 A recursive target reliability allocation algorithm for complex systems

```
allocate_sub-item_target_reliability ( ro_i )
Remark – ro_i: the parent reliability object i (a systems or system group)
Remark – The target reliability of ro_i is provided as an input to this algorithm.

Begin

ro_i,j [ ] = (ro_i).get_sub-items( );

Remark – Step 1: assign RTRWs of sub-items (ro_i,j [ ])
assign_RTRWs_of_sub-items(ro_i,j [ ]);

Remark – The second step: allocate sub-item target reliability
If (ro_i is a system) Then
      allocate_for_series[9] (ro_i, ro_i,j [ ]);
Else If (ro_i is a series system group) Then
      allocate_for_series (ro_i, ro_i,j [ ]);
Else If (ro_i is a parallel system group) Then
      allocate_for_parallel[10] (ro_i, ro_i,j [ ]);
End If

Remark – Step 2: recursively call this same function
For ( each ro_i,j )
    If (ro_i,j [j] is not a simple homogeneous system) Then
        allocate_sub-item_target_reliability (ro_i,j [j] );
    End If
End For

 End
```

5.3.3.6 Discussion

According to the relative target reliability weight approach, three representative heuristic weighting methods are defined to support the ROM method: 1) historical data-based relative target reliability weight methods, 2) design information-based relative target reliability weight method, and 3) uniform relative target reliability weight method. They are simple to use for designers in the early design stage when not much design information is available. However, when objective functions such as cost and

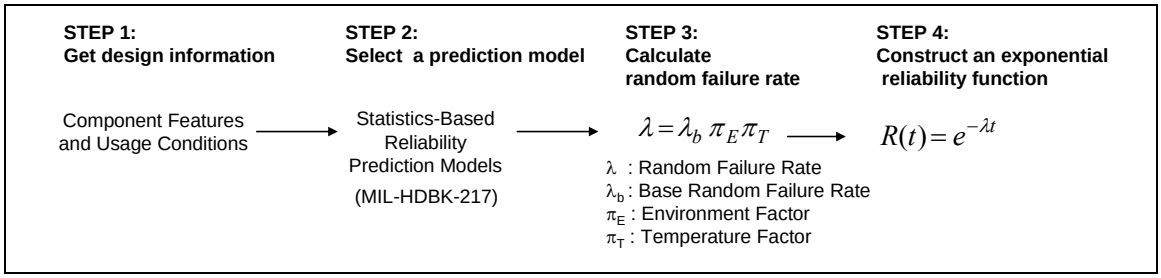
[9] Functions for target reliability allocation for series structure are described in Appendix A.
[10] Functions for target reliability allocation for parallel structure are described in Appendix A.

performance are available, other non-heuristic methods like optimization are recommended. In addition, these heuristic weighting methods are better to be applied to existing products rather than innovative products using new technologies, because expert experience is limited for the development of innovative products. Beside such limitations, to develop an effective heuristic method that captures domain expert experience correctly is future work.

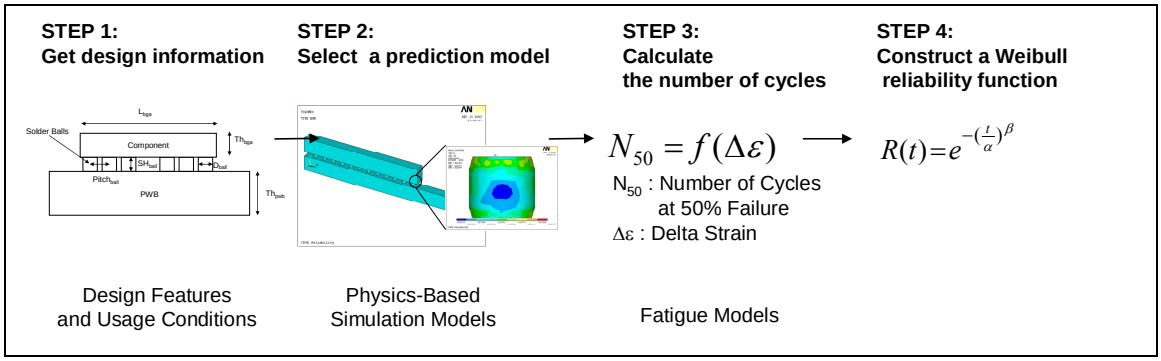
5.3.4 Reliability Prediction Algorithms

Two typical reliability prediction algorithms are illustrated in Figure 5.9. For random failure case, the first step is getting component feature and usage condition information. The second step is selecting a statistics-based prediction model, and the third step is calculating a random failure rate. The last step is constructing an exponential reliability function [Birolini, 2004; Denson, 1998] as shown in Figure 5.9-(a).

For wearout failure case, the first step is getting component feature and usage condition information [Raghunathan, 2000]. The second step is selecting a physics-based prediction model and simulating properties such as stress or strain [Tunga, 2004]. The third step is calculating the number of cycles to failure [Engelmaier, 1983]. The last step is constructing a Weibull reliability function, as shown in Figure 5.9-(b).



(a) Algorithm for predicting random failure reliability (MIL-HDBK-217 example)



(b) Algorithm for predicting wearout failure reliability (solder ball fatigue example)

Figure 5.9 Sample reliability prediction procedures

5.3.5 Reliability Assessment Algorithms

From predicted reliability data, system reliability is assessed through various reliability structures such as series, parallel, independent failure modes, and superimposable failure modes structures.

For example, the random failure rate of the system that consists of series components is shown in Equations (5.46), (5.47), and (5.48).

$$\begin{aligned} R_{r,i}(t) &= \prod_{j=1}^{n} R_{r,i.j}(t) \\ &= R_{r,i.1}(t)\, R_{r,i.2}(t) \cdots R_{r,i.n}(t) \end{aligned} \tag{5.46}$$

$$\begin{aligned} e^{-\lambda_i t} &= \prod_{j=1}^{n} e^{-\lambda_{i.j} t} \\ &= e^{-\lambda_{i.1} t}\, e^{-\lambda_{i.2} t} \cdots e^{-\lambda_{i.n} t} \\ &= e^{-(\lambda_{i.1} + \lambda_{i.2} + \cdots + \lambda_{i.n}) t} \\ &= e^{-(\sum_{j=1}^{n} \lambda_{i.j}) t} \end{aligned} \tag{5.47}$$

$$\lambda_i = \sum_{j=1}^{n} \lambda_{i.j} \tag{5.48}$$

However, no explicit formula exists for the random failure rate assessment for parallel structures. Therefore, we use the least squares method to calculate the approximated random failure rate of the system, shown in Table 5.11.

This table only shows equations for assessing random failure rates of systems and components, because we assume most failure modes are based on wearout failures, not random failures.

Table 5.11 Equations for random failure reliability assessment

Groups of systems or components			
	Series Structure		$\lambda_i = \sum_{j=1}^{n} \lambda_{i.j}$ (5.49)
	Parallel Structure		
		Homogeneous Redundancy (k-out-of-n)	$e^{-\lambda_i t} \approx \sum_{l=k}^{n} \binom{n}{l} R_{r,i.j}^{l}(t)(1-R_{r,i.j}(t))^{n-l}$ (5.50) $(0 \le t \le T_w)$
		Heterogeneous Redundancy (k-out-of-n)	$e^{-\lambda_i t} \approx \sum_{l=k}^{n} \left[\sum^{c_l^n} \{ \prod^{l} R_{i.j,selected}(t) \prod^{n-l} (1-R_{i.j,not\ selected}(t)) \} \right]$ (5.51) $(0 \le t \le T_w)$
Groups of failure modes			
	Independent Failure Modes Structure		Not Available
	Superimposable Failure Modes Structure		Not Available

i: item of interest (a system or a group),
i.j: sub-item i.j of item i (subsystems, subgroups, or components),
n: number of sub-items,
λ is the random failure rate,
$R_r(t) = e^{-\lambda t}$ is random failure reliability.

Wearout failure reliability assessment can be achieved simply by calculating basic reliability equations in Table 5.3 with target-time-to-wearout-failure (T_w). Table 5.12 shows equations for assessing wearout failure reliability.

Table 5.12 Equations for wearout failure reliability assessment

Groups of systems or components			
	Series Structure	$R_{w,i}(T_w) = \prod_{j=1}^{n} R_{w,i.j}(T_w)$	(5.52)
	Parallel Structure		
		Homogeneous Redundancy (k-out-of-n)	$R_{w,i}(T_w) = \sum_{l=k}^{n} \binom{n}{l} R_{w,i.j}^{l}(T_w)(1-R_{w,i.j}(T_w))^{n-l}$	(5.53)
		Heterogeneous Redundancy (k-out-of-n)	$R_{w,i}(T_w) = \sum_{l=k}^{n} \left[\sum^{c_l^n} \{ \prod^{l} R_{w,i.j,selected}(T_w) \prod^{n-l} (1-R_{w,i.j,not\ selected}(T_w)) \} \right]$	(5.54)
Groups of failure modes			
	Independent Failure Modes Structure	$R_{w,i.j}(T_w) = \prod_{k=1}^{m} R_{w,i.j.k}(T_w)$	(5.55)
	Superimposable Failure Modes Structure	$R_{w,i.j}(T_w) = e^{-(\frac{T_w}{\alpha})^{\beta}}$, $(\alpha,\beta) \approx f\left(N_{x_1,i.j}, N_{x_2,i.j}, ..., N_{x_n,i.j}\right)$ $Miner's\ Rule: \frac{1}{N_{x,i.j}} = \sum_{k=1}^{m} \frac{n_{i.j.k}}{N_{x,i.j.k}}$	(5.56)

i: item of interest (a system or a group),
i.j: sub-item i.j of item i (subsystems, subgroups, or components),
i.j.k: failure mode i.j.k of component i.j,
n: number of sub-items,
m: number of failure modes,
T_w is time-to-wearout-failure,
$R_W(T_W) = e^{-(\frac{T_w}{\alpha})^{\beta}}$ is wearout failure reliability at time-to-wearout-failure.

For assessing the reliabilities of complex systems that include multiple series, parallel, independent failure mode, and superimposable failure mode structures, all reliability assessment algorithms must be combined in a systematic manner from failure

modes to the top level system. This recursive assessment algorithm is described in pseudo-code form in Table 5.13.

The first step of the algorithm is to check whether the sub-item of an item is a leaf item that does not include sub-items. If the sub-item is not a leaf item, the reliability assessment function is called recursively. After all sub-items reliabilities are assessed, the reliability of the item is assessed depending on its reliability structure.

Table 5.13 A recursive reliability assessment algorithm for complex systems

```
assess_item_reliability_from_sub-item_reliabilities ( ro_i )
Remark – ro_i: reliability object i (a systems, system groups, components, or component groups)

Begin

 ro_i,j [ ] = (ro_i).get_sub-items( );

  Remark – Step 1: recursively call this same  function.
 For ( each ro_i,j  )
    If ((ro_i,j [j]).get_sub-items exist) Then
       assess_item_reliability_from_sub-item_reliabilities (ro_i,j [j]);
    End If
 End For

  Remark – Step 2: assess item reliability from sub-item reliabilities
  If (ro_i  is based on a series structure) Then
    assess_for_series[11] (ro_i, ro_i,j [ ]);
 Else If(ro_i  is based on a parallel structure) Then
    assess_for_parallel[12] (ro_i, ro_i,j [ ]);
 Else If (ro_i  is based on an independent failure model structure) Then
    assess_for_independent[13] (ro_i, ro_i,j [ ]);
 Else If(ro_i  is based on a superimposable failure model structure) Then
    assess_for_superimposable[14] (ro_i, ro_i,j [ ]);
 End If

 End
```

[11] Functions for series structure assessment are described in Appendix B.
[12] Functions for parallel structure assessment are described in Appendix B.
[13] Functions for independent failure mode structure assessment are described in Appendix B.
[14] Functions for superimposable failure mode structure assessment are described in Appendix B.

5.3.6 Design Change Recommendation Rules

If the random failure reliability of a system is less than the target random failure reliability of the system ($\lambda_{i,assessed} > \lambda_{i,target}$), design changes are requested by rules in Table 5.14. This is the first procedure for recommending design options. The second procedure entails searching for critical subsystems or components that are comparatively less reliable than others. Once critical design objects are determined, the designers then receive recommended design options resulting from expert knowledge captured in ROM. These procedures and rules are summarized in Table 5.14.

Example design recommendation rules for wearout failure reliability in the PWBA design domain are also shown in Table 5.15 and Table 5.16. The first rule set is for deciding whether a design change recommendation is necessary or not by comparing target wearout failure reliability and assessed wearout failure reliability. If so, the second rule set is used in searching for design objects that may require changes. Then, based on the types and features of the design objects, general or specific design alternatives are recommended. For example, if a PTH is a critical component that requires changes, the three design alternatives are recommended: 1) increasing PTH diameter, 2) increasing PTH thickness, and 3) changing PTH material.

Table 5.14 Sample design change recommendation rules for random failure reliability in the generic domain

Generic Rule Set R1: Determine if design changes are needed

If ($\lambda_{i,assessed} > \lambda_{i,target}$) ***Then***
call ***Generic Rule Set R2*** *and return all sub items in system i;*
End If
Remark - λ_i: random failure rate of system i

Generic Rule Set R2: Search for design objects that require changes

For *(each sub itme)*
If $\lambda_{i.j,assessed} > \lambda_{i,assessed} \times$ *criticality ratio* ***Then***
call ***Generic Rule Set R3*** *and return sub item i.j;*
End If
End For
Remark – $\lambda_{i.j}$: random failure rate of sub item i.j
Remark – *criticality ratio*: an index used to determine which sub items are critical with respect to system reliability

Generic Rule Set R3: Recommend generic design alternative type

If *(sub item i.j is a system)* ***Then***
recommend a design alternative: add redundant systems;
or
recommend a design alternative: make the system maintainable;
or
recommend a design alternative: make the system modularized;
Else If *(sub item i.j is a component)* ***Then***
recommend a design alternative: add redundant components;
or
recommend a design alternative: make the component repairable;
or
recommend a design alternative: use alternative components;
End If

Table 5.15 Sample design change recommendation rules for wearout failure reliability in the generic domain

Generic Rule Set W1: Determine if design changes are needed

If *($R_{w,i}(T_w)_{assessed} < R_{w,i}(T_w)_{target}$)* ***Then***
call ***Generic Rule Set W2*** *and return all sub items in system i;*
End If
Remark – $R_{w,i}(T_w)$: wearout failure reliability of system i

Generic Rule Set W2: Search for design objects that require changes

For *(each sub item)*
If $R_{w,i,j}(T_w)_{\ assessed} < (R_{w,i}(T_w)_{target})^{criticality\ ratio}$ ***Then***
call ***Generic Rule Set W3*** *and return sub item i.j;*
End If
End For
Remark – $R_{w,i,j}(T_w)$: wearout failure reliability of sub item i.j
Remark – *criticality ratio*: an index used to determine which sub items are critical with respect to system reliability

Generic Rule Set W3: Recommend generic design alternatives

If *(sub item i.j is a system)* ***Then***
recommend a design alternative: add redundant subsystems;
or
recommend a design alternative: make the subsystem maintainable;
or
recommend a design alternative: make the subsystem modularized;
Else If *(sub item i.j is a component)* ***Then***
recommend a design alternative: add redundant components;
or
recommend a design alternative: make the component repairable;
or
If *(component i.j has design features)* ***Then***
call ***PWBA-Domain Rule Set W1*** *and return component i.j;*
End If
End If

Table 5.16 Sample design change recommendation rules for wearout failure reliability in the PWBA domain

PWBA-Domain Rule Set W1: Recommend specific design alternatives

If *(component i.j is a solder balls (SB) interconnection)* ***Then***
recommend a design alternative: increase SB.height;
or
recommend a design alternative: increase SB.diameter;
or
recommend a design alternative: change SB.material;
Else If *(component i.j is a solder joints (SJ) interconnection)* ***Then***
recommend a design alternative: increase SJ.standoff_height;
or
recommend a design alternative: change SJ.material;
Else If *(component i.j is a plated through hole (PTH) interconnection)* ***Then***
recommend a design alternative: increase PTH.diameter;
or
recommend a design alternative: increase PTH.thickness;
or
recommend a design alternative: change PTH.material;
End If

5.4 ROM-STRUCTURE

In this section, specifications for ROM-Structure are discussed. Then, reliability classes and objects for ROM-Structure are explained.

5.4.1 Specifications for ROM-Structure

Existing failure analysis methods such as the Reliability Block Diagram (RBD) and the Fault Tree Analysis (FTA) are based exclusively on logical structure. Therefore, their capacity to support SDfR is limited because SDfR consists of three different types of structures: physical assembly structures (AS), logical structures (LS), and failure mode structures (FMS).

For example, a cell phone assembly structure is decomposed into an electronic board, a display, a battery, and a keypad. Similarly, the electronic board assembly structure is decomposed into an RF chip package, a CPU chip package, and various other chip packages. Such a multi-level physical assembly structure is illustrated in Figure 5.10.

Within a system level, multiple logical layers may exist. Three logical layers in a system are illustrated in Figure 5.11.

A component in Figure 5.10 and Figure 5.11 does not represent a unit of failure. Any component may have multiple failure modes, several of which may be either independent or superimposable. The failure mode structure is illustrated in Figure 5.12.

For the support of such structures in a unified manner, we developed a new reliability analysis structure through reliability object classification (i.e. specification or

inheritance) and composition mechanisms. This is explained in detail in the following sections.

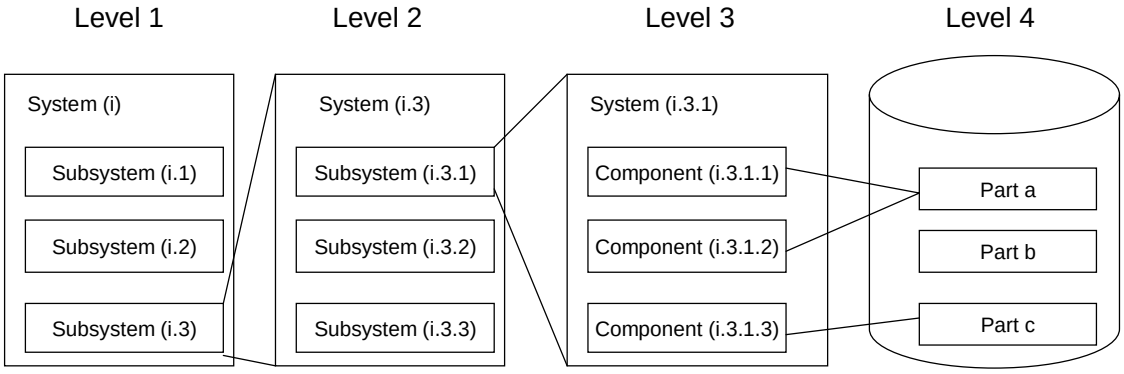


Figure 5.10 Multi-level physical assembly structures

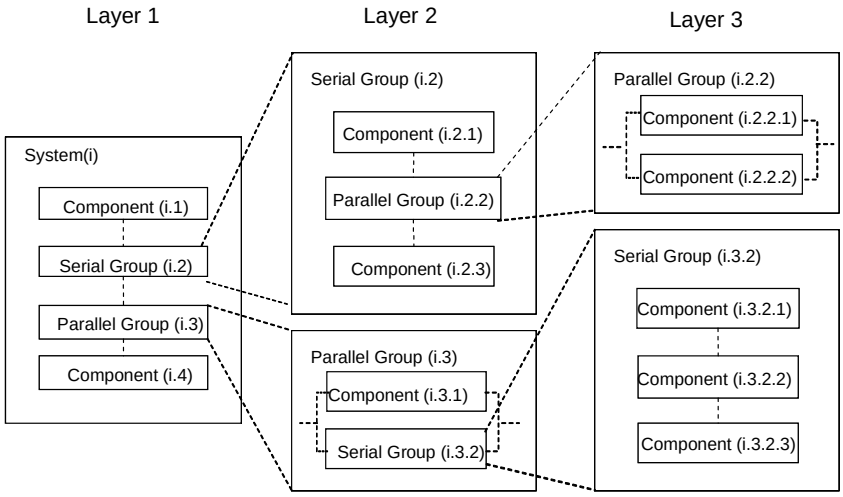


Figure 5.11 Multi-layer logical structures

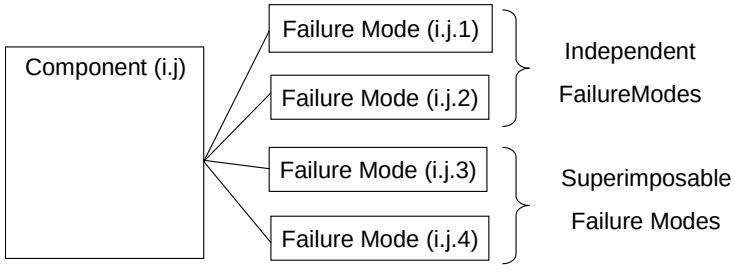
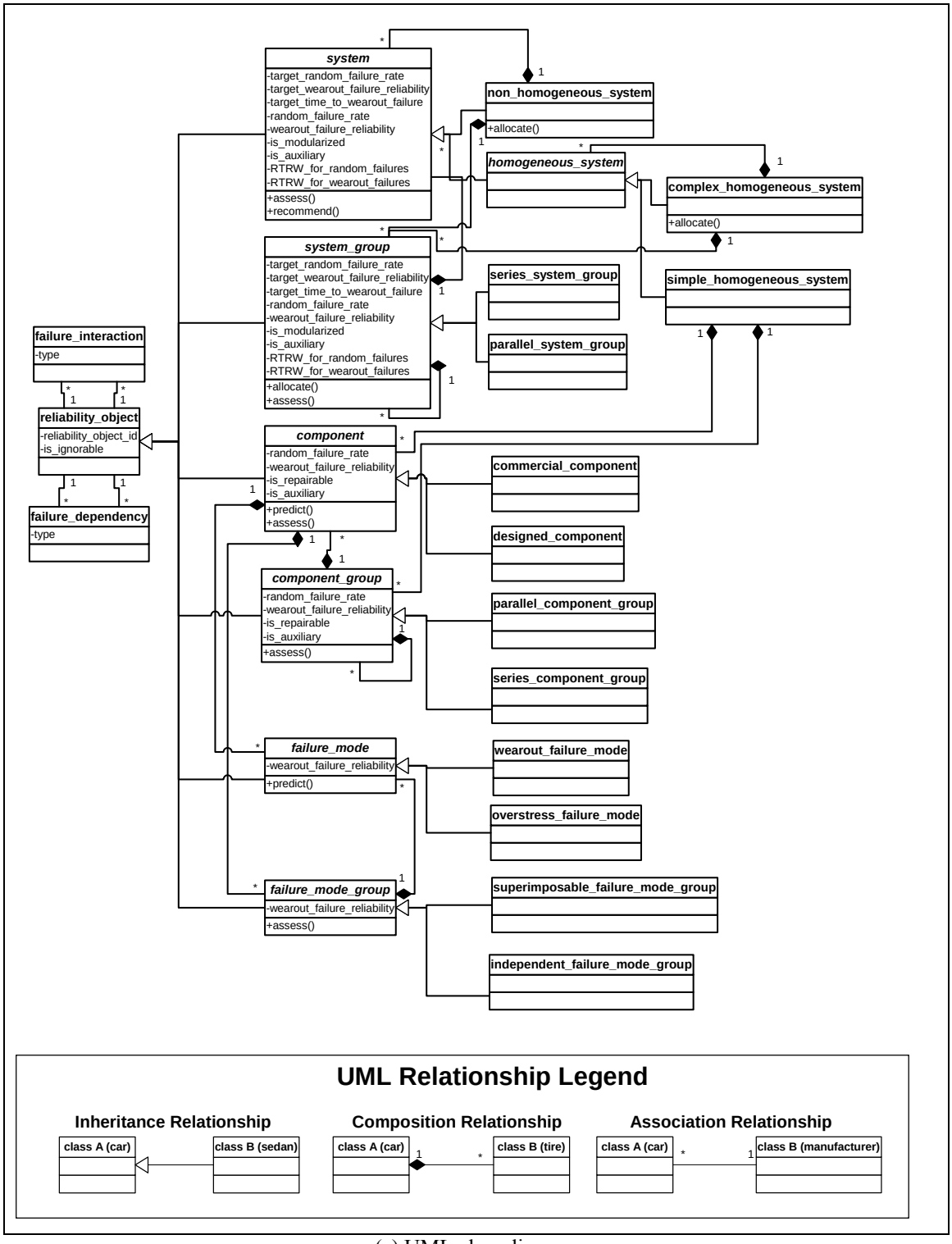


Figure 5.12 Multi-mode structures

5.4.2 Reliability Classes for ROM-Structure



(a) UML class diagram

Figure 5.13 A UML class diagram for ROM-Structure (Continue)

- **System**: A physical collection of subsystems or components organized to accomplish a specific function or set of functions [IEEE, 1990].
- **Non-homogeneous system**: A system that includes multi-disciplinary aspects.
- **Homogeneous System**: A system that includes only uni-disciplinary aspects.
- **Complex Homogeneous System**: A homogeneous system that consists of homogeneous subsystems. A subsystem is a set of components in the whole system which is a system itself [Webster's College Dictionary, 1997]
- **Simple Homogeneous System**: A homogenous system that does not include any subsystems, but only components.
- **Component**: A usage of a part in an assembly. A part is a reusable black box used in assembling systems [Webster's College Dictionary, 1997].
- **Failure Mode**: A characterization of the way a component fails [Jensen, 1995].

(b) Definition summaries

Figure 5.13 A UML class diagram for ROM-Structure

A new unified reliability analysis structure, called ROM-Structure, is designed and illustrated in UML Class Diagram in Figure 5.13. The diagram consists of the classification (i.e. specification or inheritance) and the composition relationships among reliability objects.

The classification starts with the *reliability object* class. It is the super class of most of the objects that comprise ROM-Structure. The *reliability object* class is specialized into six basic classes: *system*, *system group*, *component*, *component group*, *failure mode*, and *failure mode group* for supporting AS, LS, and FMS. Then, for the complete support of SDfR, these basic classes are again specialized into fourteen subclasses making 21 classes in all. *System* is specialized into the subclasses *non-homogeneous system*, *homogeneous system*, *complex homogeneous system*, and *simple homogeneous system* for supporting multi-level and multi-disciplinary concepts; *system group* is specialized into the subclasses *series system group* and *parallel system group* for

supporting statistic principles*; component* is specialized into the subclasses *commercial component* and *designed component* for supporting different types of component design in a system context; *component group* is specialized into the subclasses *parallel component group* and *series component group* for supporting statistic principles; *failure mode* is specialized into the subclasses *wearout failure mode* and *overstress failure mode* for supporting different types of failure modes; and *failure mode group* is specialized into the subclasses *superimposable failure mode* and *independent failure mode* for supporting physics principles. Usage of these classes is described in detail with examples in Chapter 7.

Based on this classification, composition relationships are modeled for representing the hierarchical nature of ROM-Structure as a tree structure. For example, a non-homogeneous system object is composed of system objects or system group objects. These composition relationships are modeled by lines with black diamonds that branch out from the *non-homogeneous system* class in Figure 5.13. Other composition relationships for other classes are also illustrated in Figure 5.13. The composition relationships indicate rules for available child node types in instance tree structures, while the classification indicates various node types in instance tree structures.

In addition to the 21 classes that construct instance tree structures, two relational classes are presented — *failure interaction* and *failure dependency* (Figure 5.13). The *failure interaction* class consists of components or subsystems that are considered to mutually affect system failures. For example, electronic systems may fail by electromagnetic interference (EMI) among electrical circuit boards that emit high frequency electromagnetic radiation. Another example is high temperature-induced

failure of electronic systems. This may be due to the presence of two or more high power packages assembled in close proximity to each other. These possible failures are captured in a ROM model as failure interaction instances that consist of relevant reliability objects and the description of failure interaction type at design stages. Thus, such interactions are checked by simulations or prototype testing.

The *failure dependency* class consists of components and subsystems where the reliability of one item is considered to be limited by the reliability of a different item. For example, the reliability of a microprocessor on a motherboard can be considered to be dependent on the reliability of the system fan. This dependency is captured during the design stage in a ROM model as a failure dependency instance that consists of relevant reliability objects and a description of the failure dependency. Thus, such dependencies should be considered for system design for reliability. Usage of this class is described in detail with examples in Chapter 7.

Options for reliability importance [Leemis, 1995] and maintenance are also defined as class attributes in ROM-Structure. For example, *system* class includes *is_modularized* and *is_auxiliary* attributes in Figure 5.13.

The details of each reliability object are described in the following section.

5.4.3 Reliability Objects for ROM-Structure

5.4.3.1 System Objects

According to [IEEE, 1990], a system is defined as follows:

Definition 5-4

*A **System** is a physical collection of subsystems or components organized to accomplish a specific function or set of functions.*

Systems can be specialized by multi-disciplinary characteristics. A system that consists of multi-disciplinary subsystems is called ***nonhomogeneous system***, and a system that consists of single disciplinary subsystems is called ***homogeneous system***. They are defined as follows:

Definition 5-5

*A **Nonhomogeneous System** is a system that includes multi-disciplinary aspects.*

Definition 5-6

*A **Homogeneous System** is a system that includes only a single disciplinary aspect.*

Based on the definition of nonhomogeneous systems, a reliability object for nonhomogeneous systems is represented in mathematical form in Equation (5.57). The nonhomogeneous system object consists of five reliability metrics (i.e. λ_{target}, T_w, $R_w(T_w)_{target}$, $\lambda_{assessed}$, and $R_w(T_w)_{assessed}$), three reliability activities (i.e. $RA_{allocate\_for\_series}$,

$RA_{assess\_for\_series}$, and $RA_{recommend}$), and composition relationships with $\mathbf{ro_{system,i}}$ and $\mathbf{ro_{system\_group,i}}$.

$$\mathbf{ro_{nh\_system,i}} \equiv (\mathbf{roid}, \mathbf{RM_{nh\_system}}, \mathbf{RA_{nh\_system}}, \mathbf{RS_{nh\_system}}) \quad (5.57)$$

where

$\mathbf{ro_{nh\_system,i}}$ is a nonhomogeneous system object.

$\mathbf{RM_{nh\_system}}$ consists of five elements ($\mathbf{RM_{nh\_system}}$ = {(λ_{target}, real, null), (T_w, real, null), ($R_w(T_w)_{target}$, probability, null), ($\lambda_{assessed}$, real, null), ($R_w(T_w)_{assessed}$, probability, null)}).

$\mathbf{RA_{nh\_system}}$ consists of three elements ($\mathbf{RA_{nh\_system}}$ = {($RA_{allocate\_for\_series}$, null), ($RA_{assess\_for\_series}$, null), ($RA_{recommend}$, null)}).

$\mathbf{RS_{nh\_system}}$ are composition relationships among $\mathbf{ro_{nh\_system,i}}$ and other reliability objects that construct reliability structure. The sub reliability objects are $\mathbf{ro_{system,i}}$, $\mathbf{ro_{system\_group,i}}$.

Furthermore, homogeneous systems are specialized by multi-level characteristics. A homogeneous system that consists of subsystems is called ***complex homogeneous system***, and a homogeneous system that includes only components but not any subsystems is called ***simple homogeneous system***. They are defined as follows:

Definition 5-7

*A **Complex Homogeneous System** is a system that consists of homogeneous subsystems.*

Definition 5-8

*A **Simple Homogeneous System** is a system that includes only components, but not any subsystem.*

Based on these definitions, reliability objects for complex homogeneous systems and simple homogeneous systems are represented in mathematical form in Equations (5.58) and (5.59). The complex homogeneous system object consists of five reliability metrics (i.e. λ_{target}, T_w, $R_w(T_w)_{target}$, $\lambda_{assessed}$, and $R_w(T_w)_{assessed}$), three reliability activities (i.e. $RA_{allocate\_for\_series}$, $RA_{assess\_for\_series}$, and $RA_{recommend}$), and composition relationships with $\mathbf{ro}_{\mathbf{h\_system,i}}$ and $\mathbf{ro}_{\mathbf{system\_group,i}}$. The simple homogeneous system object consists of five reliability metrics (i.e. λ_{target}, T_w, $R_w(T_w)_{target}$, $\lambda_{assessed}$, and $R_w(T_w)_{assessed}$), two reliability activities (i.e. $RA_{assess\_for\_series}$ and $RA_{recommend}$), and composition relationships with $\mathbf{ro}_{\mathbf{component,i}}$ and $\mathbf{ro}_{\mathbf{component\_group,i}}$.

$$\mathbf{ro}_{\mathbf{c\_system,i}} \equiv (\mathbf{roid}, \mathbf{RM}_{\mathbf{c\_system}}, \mathbf{RA}_{\mathbf{c\_system}}, \mathbf{RS}_{\mathbf{c\_system}}) \quad (5.58)$$

where

$\mathbf{ro}_{\mathbf{c\_system,i}}$ is a complex homogeneous system object.

$\mathbf{RM}_{\mathbf{c\_system}}$ consists of five elements ($\mathbf{RM}_{\mathbf{c\_system}}$ = {(λ_{target}, real, null), (T_w, real, null), ($R_w(T_w)_{target}$, probability, null), ($\lambda_{assessed}$, real, null), ($R_w(T_w)_{assessed}$, probability, null)}).

$\mathbf{RA}_{\mathbf{c\_system}}$ consists of three elements ($\mathbf{RA}_{\mathbf{c\_system}}$ = {($RA_{allocate\_for\_series}$, null), ($RA_{assess\_for\_series}$, null), ($RA_{recommend}$, null)}).

$\mathbf{RS}_{\mathbf{c\_system}}$ are composition relationships among $\mathbf{ro}_{\mathbf{nh\_system,i}}$ and other reliability objects that construct reliability structure. The sub reliability objects are $\mathbf{ro}_{\mathbf{h\_system,i}}$, $\mathbf{ro}_{\mathbf{system\_group,i}}$.

$$\mathbf{ro}_{\mathbf{s\_system,i}} \equiv (\mathbf{roid}, \mathbf{RM}_{\mathbf{s\_system}}, \mathbf{RA}_{\mathbf{s\_system}}, \mathbf{RS}_{\mathbf{s\_system}}) \quad (5.59)$$

where

$\mathbf{ro}_{\mathbf{s\_system,i}}$ is a simple homogeneous system object.

$\mathbf{RM}_{\mathbf{s\_system}}$ consists of five elements ($\mathbf{RM}_{\mathbf{s\_system}}$ = { (λ_{target}, real, null), (T_w, real, null), ($R_w(T_w)_{target}$, probability, null), ($\lambda_{assessed}$, real, null), ($R_w(T_w)_{assessed}$, probability, null)}).

$\mathbf{RA}_{\mathbf{s\_system}}$ consists of two elements ($\mathbf{RA}_{\mathbf{s\_system}}$ = {($RA_{assess\_for\_series}$, null), ($RA_{recommend}$, null)}).

$\mathbf{RS}_{\mathbf{s\_system}}$ are composition relationships among $\mathbf{ro}_{\mathbf{s\_system,i}}$ and other reliability objects that construct reliability structure. The sub reliability objects are $\mathbf{ro}_{\mathbf{component,i}}$ and $\mathbf{ro}_{\mathbf{component\_group,i}}$.

These three subtypes of system objects (i.e. *nonhomogeneous system object*, *complex homogeneous system object*, and *simple homogeneous system object*) can represent most electronic packaging systems. For examples, a hard disk drive that consists of an electronic board and a mechanical arm-and-head system is a nonhomogeneous system; a main board assembly in a note book PC that consists of multiple electronic boards is a complex homogeneous system; and an electronic board that consists of multiple electronic components is a simple homogeneous system.

However, these examples are only valid for note book PC designers, but not for microprocessor designers. For microprocessor designers, a microprocessor is a complex homogenous system, but not a component. Therefore, classifying systems is depending on the context of system design.

In addition to the specialization by the multi-disciplinary and multi-level characteristics, systems are categorized into three types for reliability calculation: a

modularized system, an ***ignorable system***, and an ***auxiliary system***. The meanings of these types are described in Table 5.17. Each type is also represented by a Boolean type.

Table 5.17 System object types

Type	Description
Modularized System	The system is easily interchangeable and has its own target reliability.
Ignorable System	The system is considered either irrelevant to the main functionalities of the top level system or highly reliable compared with other items, so it can be ignored when target reliability is allocated or reliability is predicted and assessed.
Auxiliary System	The system is not directly involved in the main functionality, but it may influence the reliability of other systems.

5.4.3.2 System Group Objects

We define a system group as follows:

Definition 5-9

*A **System Group** is a logical group of subsystems or subsystem groups.*

System groups can be specialized into two logical structure types: a series structure type and a parallel structure type. A system group that consists of subsystems or subsystem groups with a series structure in terms of failures is called ***series system group***, and a system group that consists of subsystems or subsystem groups with a parallel structure in terms of failures is called ***parallel system group***. They are defined as follows:

Definition 5-10

*A **Series System Group** is a system group that consists of subsystems or subsystem groups with a series structure in terms of failures.*

Definition 5-11

*A **Parallel System Group** is a system group that consists of subsystems or subsystem groups with a parallel structure in terms of failures.*

Based on these definitions, reliability objects for series system groups and parallel system groups are represented in mathematical form in Equations (5.60) and (5.61). The series system group object consists of five reliability metrics (i.e. λ_{target}, T_w, $R_w(T_w)_{target}$, $\lambda_{assessed}$, and $R_w(T_w)_{assessed}$), two reliability activities (i.e. $RA_{allocate\_for\_series}$ and $RA_{assess\_for\_series}$), and composition relationships with $\mathbf{ro}_{\mathbf{system,i}}$ and $\mathbf{ro}_{\mathbf{system\_group,i}}$. The parallel system group object consists of five reliability metrics (i.e. λ_{target}, T_w, $R_w(T_w)_{target}$, $\lambda_{assessed}$, and $R_w(T_w)_{assessed}$), two reliability activities (i.e. $RA_{allocate\_for\_parallel}$ and $RA_{assess\_for\_parallel}$), and composition relationships with $\mathbf{ro}_{\mathbf{system,i}}$ and $\mathbf{ro}_{\mathbf{system\_group,i}}$.

$$\mathbf{ro}_{\mathbf{s\_system\_group,i}} \equiv (\mathbf{roid}, \mathbf{RM}_{\mathbf{s\_system\_group}}, \mathbf{RA}_{\mathbf{s\_system\_group}}, \mathbf{RS}_{\mathbf{s\_system\_group}}) \quad (5.60)$$

where

$\mathbf{ro}_{\mathbf{s\_system\_group,i}}$ is a series system group object.

$\mathbf{RM}_{\mathbf{s\_system\_group}}$ consists of five elements ($\mathbf{RM}_{\mathbf{s\_system\_group}}$ = {(λ_{target}, real, null), (T_w, real, null), ($R_w(T_w)_{target}$, probability, null), ($\lambda_{assessed}$, real, null), ($R_w(T_w)_{assessed}$, probability, null)}).

$\mathbf{RA}_{\mathbf{s\_system\_group}}$ consists of two elements ($\mathbf{RA}_{\mathbf{s\_system\_group}}$ = {($RA_{allocate\_for\_series}$, null), ($RA_{assess\_for\_series}$, null)).

$\mathbf{RS}_{\mathbf{s\_system\_group}}$ are composition relationships among $\mathbf{ro}_{\mathbf{s\_system\_group,i}}$ and other reliability objects. The sub reliability objects are $\mathbf{ro}_{\mathbf{system,i}}$ and $\mathbf{ro}_{\mathbf{system\_group,i}}$.

$$\mathbf{ro}_{\mathbf{p\_system\_group,i}} \equiv (\mathbf{roid}, \mathbf{RM}_{\mathbf{p\_system\_group}}, \mathbf{RA}_{\mathbf{p\_system\_group}}, \mathbf{RS}_{\mathbf{p\_system\_group}}) \quad (5.61)$$

where

$\mathbf{ro}_{\mathbf{p\_system\_group,i}}$ is a parallel system group object.

$\mathbf{RM}_{\mathbf{p\_system\_group}}$ consists of five elements ($\mathbf{RM}_{\mathbf{p\_system\_group}}$ = {(λ_{target}, real, null), (T_w, real, null), ($R_w(T_w)_{target}$, probability, null), ($\lambda_{assessed}$, real, null), ($R_w(T_w)_{assessed}$, probability, null)}).

$\mathbf{RA}_{\mathbf{p\_system\_group}}$ consists of two elements ($\mathbf{RA}_{\mathbf{p\_system\_group}}$ = {($RA_{allocate\_for\_parallel}$, null), ($RA_{assess\_for\_parallel}$, null)).

$\mathbf{RS}_{\mathbf{p\_system\_group}}$ are composition relationships among $\mathbf{ro}_{\mathbf{p\_system\_group,i}}$ and other reliability objects that construct reliability structure. The sub reliability objects are $\mathbf{ro}_{\mathbf{system,i}}$ and $\mathbf{ro}_{\mathbf{system\_group,i}}$.

A group of input subsystems in a notebook PC (e.g. a keyboard, a mouse, and a touch pad) is an example of a series system group. A Redundant Array of Independent Drives (RAID) that consists of multiple backup disks is an example of a parallel system group.

System groups are categorized into three types for reliability calculation: a ***modularized system group***, an ***ignorable system group***, and an ***auxiliary system group***. These types are described in Table 5.18.

Table 5.18 System group object types

Type	Description
Modularized System Group	The system group is easily interchangeable and has its own target reliability.
Ignorable System Group	The system group is considered either irrelevant to the main functionalities of the top level system or highly reliable compared with other items, so it can be ignored when target reliability is allocated or reliability is predicted and assessed.
Auxiliary System Group	The system group is not directly involved in the main functionality, but it may influence the reliability of other systems.

5.4.3.3 Component Objects

According to [Webster's College Dictionary, 1997], a component is defined as follows:

Definition 5-12

*A **Component** is a usage of a part in a system. A part is a reusable item in a library for system design.*

Components can be specialized into two different component design types. A component that is laid out by selecting a commercial part is called ***commercial component***, and a component that is designed by using a set of parameters (a feature) is called ***designed component***. They are defined as follows:

Definition 5-13

*A **Commercial Component** is a component that is laid out by selecting a commercial part in a part library.*

Definition 5-14

*A **Designed Component** is a component that is designed by using a set of parameters (a feature).*

Based on these definitions, reliability objects for commercial components and design components are represented in mathematical form in Equations (5.62) and (5.63). The commercial component object consists of two reliability metrics (i.e. $\lambda_{predicted}$ and $R_w(T_w)_{predicted}$), two reliability activities (i.e. $RA_{predict}$ and RA_{assess}), and composition relationships with $\mathbf{ro}_{\mathbf{failure\_mode,i}}$ and $\mathbf{ro}_{\mathbf{failure\_mode\_group,i}}$. The designed component object consists of two reliability metrics (i.e. $\lambda_{predicted}$ and $R_w(T_w)_{predicted}$), two reliability

activities (i.e. $RA_{predict}$ and RA_{assess}), and composition relationships with $\mathbf{ro}_{\mathbf{failure\_mode,i}}$ and $\mathbf{ro}_{\mathbf{failure\_mode\_group,i}}$.

$$\mathbf{ro}_{\mathbf{c\_component,i}} \equiv (\mathbf{roid}, \mathbf{RM}_{\mathbf{c\_component}}, \mathbf{RA}_{\mathbf{c\_component}}, \mathbf{RS}_{\mathbf{c\_component}}) \quad (5.62)$$

where

$\mathbf{ro}_{\mathbf{c\_component,i}}$ is a commercial component object.

$\mathbf{RM}_{\mathbf{c\_component}}$ consists of two elements ($\mathbf{RM}_{\mathbf{c\_component}}$ = {($\lambda_{predicted}$, real, null), ($R_w(T_w)_{predicted}$, probability, null)}).

$\mathbf{RA}_{\mathbf{c\_component}}$ consists of two elements ($\mathbf{RA}_{\mathbf{c\_component}}$ = {($RA_{predict}$, null), (RA_{assess}, null)}).

$\mathbf{RS}_{\mathbf{c\_component}}$ are composition relationships among $\mathbf{ro}_{\mathbf{c\_component,i}}$ and other reliability objects that construct reliability structure. The sub reliability objects are $\mathbf{ro}_{\mathbf{failure\_mode,i}}$ and $\mathbf{ro}_{\mathbf{failure\_mode\_group,i}}$.

$$\mathbf{ro}_{\mathbf{d\_component,i}} \equiv (\mathbf{roid}, \mathbf{RM}_{\mathbf{d\_component}}, \mathbf{RA}_{\mathbf{d\_component}}, \mathbf{RS}_{\mathbf{d\_component}}) \quad (5.63)$$

where

$\mathbf{ro}_{\mathbf{d\_component,i}}$ is a designed component object.

$\mathbf{RM}_{\mathbf{d\_component}}$ consists of two elements ($\mathbf{RM}_{\mathbf{d\_component}}$ = {($\lambda_{predicted}$, real, null), ($R_w(T_w)_{predicted}$, probability, null)}).

$\mathbf{RA}_{\mathbf{d\_component}}$ consists of two elements ($\mathbf{RA}_{\mathbf{d\_component}}$ = {($RA_{predict}$, null), (RA_{assess}, null)}).

$\mathbf{RS}_{\mathbf{d\_component}}$ are composition relationships among $\mathbf{ro}_{\mathbf{d\_component,i}}$ and other reliability objects that construct reliability structure. The sub reliability objects are $\mathbf{ro}_{\mathbf{failure\_mode,i}}$ and $\mathbf{ro}_{\mathbf{failure\_mode\_group,i}}$.

Resistors and capacitors in an electronic board system are examples of commercial components. Various interconnections such as solder joints and plated through holes in an electronic board system are examples of designed components.

Components are categorized into three types for reliability calculation: a ***repairable component***, an ***ignorable component***, and an ***auxiliary component***. The meanings of these types are described in Table 5.19.

Table 5.19 Component object types

Type	Description
Repairable Component	The component is easily repaired or replaced.
Ignorable Component	The component is considered either irrelevant to the main functionalities of the top level system or highly reliable compared with other items, so it can be ignored when reliability is predicted and assessed.
Auxiliary Component	The component is not directly involved in the main functionality, but it may influence the reliability of other components.

5.4.3.4 Component Group Objects

We define a component group as follows:

Definition 5-15

*A **Component Group** is a logical group of components or component groups.*

Component group is specialized into two logical structure types: a series structure type and a parallel structure type. A component group that consists of components or component groups with a series structure in terms of failures is called ***series component group***, and a component that consists of components or component groups with a parallel structure in terms of failures is called ***parallel component group***. They are defined as follows:

Definition 5-16

*A **Series Component Group** is a component group that consists of components or component groups with a series structure in terms of failures.*

Definition 5-17

*A **Parallel Component Group** is a component group that consists of components or component groups with a parallel structure in terms of failures.*

Based on these definitions, reliability objects for series component groups and parallel component groups are represented in mathematical form in Equations (5.64) and (5.65). The series component group object consists of two reliability metrics (i.e. $\lambda_{assessed}$, and $R_w(T_w)_{assessed}$), one reliability activity (i.e. $RA_{assess\_for\_series}$), and composition relationships with $\mathbf{ro}_{\mathbf{component,i}}$ and $\mathbf{ro}_{\mathbf{component\_group,i}}$. The parallel component group object consists of two reliability metrics (i.e. $\lambda_{assessed}$, and $R_w(T_w)_{assessed}$), one reliability activity (i.e. $RA_{assess\_for\_parallel}$), and composition relationships with $\mathbf{ro}_{\mathbf{component,i}}$ and $\mathbf{ro}_{\mathbf{component\_group,i}}$.

$$\mathbf{ro}_{\mathbf{s\_component\_group,i}} \equiv (\mathbf{roid}, \mathbf{RM}_{\mathbf{s\_component\_group}}, \mathbf{RA}_{\mathbf{s\_component\_group}}, \mathbf{RS}_{\mathbf{s\_component\_group}}) \quad (5.64)$$

where

$\mathbf{ro}_{\mathbf{s\_component\_group,i}}$ is a series component group object.

$\mathbf{RM}_{\mathbf{s\_component\_group}}$ consists of two elements ($\mathbf{RM}_{\mathbf{s\_component\_group}}$ = {($\lambda_{assessed}$, real, null), ($R_w(T_w)_{assessed}$, probability, null)}).

$\mathbf{RA}_{\mathbf{s\_component\_group}}$ consists of one element ($\mathbf{RA}_{\mathbf{s\_component\_group}}$ = {($RA_{assess\_for\_series}$, null)).

$\mathbf{RS}_{\mathbf{s\_component\_group}}$ are composition relationships among $\mathbf{ro}_{\mathbf{s\_component\_group,i}}$ and other reliability objects that construct reliability structure. The sub reliability objects are $\mathbf{ro}_{\mathbf{component,i}}$ and $\mathbf{ro}_{\mathbf{component\_group,i}}$.

$$\mathbf{ro}_{\mathbf{p\_component\_group,i}} \equiv (\mathbf{roid}, \mathbf{RM}_{\mathbf{p\_component\_group}}, \mathbf{RA}_{\mathbf{p\_component\_group}}, \mathbf{RS}_{\mathbf{p\_component\_group}}) \quad (5.65)$$

where

$\mathbf{ro}_{\mathbf{p\_component\_group,i}}$ is a parallel component group object.

$\mathbf{RM}_{\mathbf{p\_component\_group}}$ consists of two elements ($\mathbf{RM}_{\mathbf{p\_component\_group}}$ = {($\lambda_{assessed}$, real, null), ($R_w(T_w)_{assessed}$, probability, null)}).

$\mathbf{RA}_{\mathbf{p\_component\_group}}$ consists of one element ($\mathbf{RA}_{\mathbf{p\_component\_group}}$ = {($RA_{assess\_for\_series}$, null)).

$\mathbf{RS}_{\mathbf{p\_component\_group}}$ are composition relationships among $\mathbf{ro}_{\mathbf{p\_component\_group,i}}$ and other reliability objects that construct reliability structure. The sub reliability objects are $\mathbf{ro}_{\mathbf{component,i}}$ and $\mathbf{ro}_{\mathbf{component\_group,i}}$.

A group of components that implements a function of a system is an example of a series component group. A group of components that implements redundancy in a system is an example of a parallel component group.

Component groups are categorized into three types for reliability calculation: a ***repairable component group***, an ***ignorable component group***, and an ***auxiliary component group***. The meanings of these types are described in Table 5.20.

Table 5.20 Component group object types

Type	Description
Repairable Component Group	The component group is easily repaired or replaced.
Ignorable Component Group	The component group is considered either irrelevant to the main functionalities of the top level system or highly reliable compared with other items, so it can be ignored when reliability is assessed.
Auxiliary Component Group	The component group is not directly involved in the main functionality, but it may influence the reliability of other components.

5.4.3.5 Failure Mode Objects

According to [Jensen, 1995], a failure mode is defined as follows:

Definition 5-18

*A **Failure Mode** is a specific way of component fails under a specific load.*

Failure modes are specialized into two different failure mode types: a wearout failure mode type and a overstress failure mode type. They are defined as follows:

Definition 5-19

*A **Wearout Failure Mode** is a progressively deteriorated failure of a component under a specific load.*

Definition 5-20

*An **Overstress Failure Mode** is an abrupt component failure under an excessive load.*

Based on these definitions, reliability objects for wearout failure modes and overstress failure modes are represented in mathematical form in Equations (5.66) and (5.67). The wearout failure mode object consists of one reliability metric (i.e. $(R_w(T_w)_{predicted})$ and one reliability activity (i.e. $RA_{predicted\_Rw}$). The overstress failure mode object consists of one reliability metric (i.e. $(R_w(T_w)_{predicted})$ and one reliability activity (i.e. $RA_{predicted\_Rw}$).

$$\mathbf{ro}_{\mathbf{w\_failure\_mode,i}} \equiv (\mathbf{roid}, \mathbf{RM}_{\mathbf{w\_failure\_mode}}, \mathbf{RA}_{\mathbf{w\_failure\_mode}}) \quad (5.66)$$

where

$\mathbf{ro}_{\mathbf{w\_failure\_mode,i}}$ is a wearout failure mode object.

$\mathbf{RM}_{\mathbf{w\_failure\_mode}}$ consists of one element ($\mathbf{RM}_{\mathbf{w\_failure\_mode}}$ = {($R_w(T_w)_{predicted}$, probability, null)}).

$\mathbf{RA}_{\mathbf{w\_failure\_mode}}$ consists of one element ($\mathbf{RA}_{\mathbf{w\_failure\_mode}}$ = {($RApredicted_{\_Rw}$, null)}).

$$\mathbf{ro}_{\mathbf{o\_failure\_mode,i}} \equiv (\mathbf{roid}, \mathbf{RM}_{\mathbf{o\_failure\_mode}}, \mathbf{RA}_{\mathbf{o\_failure\_mode}}) \quad (5.67)$$

where

$\mathbf{ro}_{\mathbf{o\_failure\_mode,i}}$ is a wearout failure mode object.

$\mathbf{RM}_{\mathbf{o\_failure\_mode}}$ consists of one element ($\mathbf{RM}_{\mathbf{o\_failure\_mode}}$ = {($R_w(T_w)_{predicted}$, probability, null)}).

$\mathbf{RA}_{\mathbf{o\_failure\_mode}}$ consists of one element ($\mathbf{RA}_{\mathbf{o\_failure\_mode}}$ = {($RApredicted_{\_Rw}$, null) }).

A thermo-mechanical fatigue failure of solder joints is an example of a wearout failure mode. An Electro-Static Discharge (ESD) failure of capacitors is an example of an overstress failure mode.

Some failure modes are ignored for reliability calculation. This type is described in Table 5.21.

Table 5.21 Failure mode object types

Type	Description
Ignorable Failure Mode	The failure mode is considered highly reliable, so it can be ignored when reliability is predicted.

5.4.3.6 Failure Mode Group Objects

We define a failure mode group as follows:

Definition 5-21

*A **Failure Mode Group** is a logical group of failure modes.*

Failure mode groups are specialized by two different structures: an independent failure mode structure and a superimposable failure mode structure. A failure mode group that consists of independent failure modes is called ***independent failure mode group***, and a failure mode group that consists of superimposable failure modes is called ***superimposable failure mode group***. They are defined as follows:

Definition 5-22

*An **Independent Failure Mode Group** is a failure mode group of independent failure modes.*

Definition 5-23

*A **Superimposable Failure Mode Group** is a failure mode group of superimposable failure modes.*

Based on these definitions, reliability objects for independent failure mode groups and superimposable failure mode groups are represented in mathematical form in Equations (5.68) and (5.69). The independent failure mode group object consists of one reliability metric (i.e. $R_w(T_w)_{assessed}$), one reliability activity (i.e. $RA_{assess\_Rw\_for\_ind}$), and a composition relationship with $\mathbf{ro}_{\mathbf{failure\_mode,i}}$. The superimposable failure mode group

object consists of one reliability metric (i.e. $R_w(T_w)_{assessed}$), one reliability activity (i.e. $RA_{assess\_Rw\_for\_sup}$), and a composition relationship with $\mathbf{ro}_{\mathbf{failure\_mode,i}}$.

$$\mathbf{ro}_{\mathbf{ind\_failure\_mode\_group,i}} \equiv (\mathbf{roid}, \mathbf{RM}_{\mathbf{ind\_failure\_mode\_group}}, \mathbf{RA}_{\mathbf{ind\_failure\_mode\_group}}, \mathbf{RS}_{\mathbf{ind\_failure\_mode\_group}}) \quad (5.68)$$

where

$\mathbf{ro}_{\mathbf{ind\_failure\_mode\_group,i}}$ is an independent failure mode group object.

$\mathbf{RM}_{\mathbf{ind\_failure\_mode\_group}}$ consists of one element ($\mathbf{RM}_{\mathbf{ind\_failure\_mode\_group}}$ = {($R_w(T_w)_{assessed}$, probability, null)}).

$\mathbf{RA}_{\mathbf{ind\_failure\_mode\_group}}$ consists of one element ($\mathbf{RA}_{\mathbf{ind\_failure\_mode\_group}}$ = {($RA_{assess\_Rw\_for\_ind}$, null)).

$\mathbf{RS}_{\mathbf{ind\_failure\_mode\_group}}$ are composition relationships among $\mathbf{ro}_{\mathbf{ind\_failure\_mode\_group,i}}$ and other reliability objects that construct reliability structure. The sub reliability objects are $\mathbf{ro}_{\mathbf{failure\_mode,i}}$.

$$\mathbf{ro}_{\mathbf{sup\_failure\_mode\_group,i}} \equiv (\mathbf{roid}, \mathbf{RM}_{\mathbf{sup\_failure\_mode\_group}}, \mathbf{RA}_{\mathbf{sup\_failure\_mode\_group}}, \mathbf{RS}_{\mathbf{sup\_failure\_mode\_group}}) \quad (5.69)$$

where

$\mathbf{ro}_{\mathbf{sup\_failure\_mode\_group,i}}$ is a superimposable failure mode group object.

$\mathbf{RM}_{\mathbf{sup\_failure\_mode\_group}}$ consists of one element ($\mathbf{RM}_{\mathbf{sup\_failure\_mode\_group}}$ = {($R_w(T_w)_{assessed}$, probability, null)}).

$\mathbf{RA}_{\mathbf{sup\_failure\_mode\_group}}$ consists of one element ($\mathbf{RA}_{\mathbf{sup\_failure\_mode\_group}}$ = {($RA_{assess\_Rw\_for\_sup}$, null)).

$\mathbf{RS}_{\mathbf{sup\_failure\_mode\_group}}$ are composition relationships among $\mathbf{ro}_{\mathbf{sup\_failure\_mode\_group,i}}$ and other reliability objects that construct reliability structure. The sub reliability objects are $\mathbf{ro}_{\mathbf{failure\_mode,i}}$.

A thermo-mechanical fatigue failure of solder joints and a diffusion failure of solder joints are regarded as independent, so they make an independent failure mode group. If thermal cycling and mechanical vibration loads are applied to solder joints at

the same time, the failure of solder joints will be accelerated compared with the failure by each load. Therefore, the thermo-mechanical fatigue failure and the vibration-induced fatigue failure are regarded as superimposable, so they make a superimposable failure mode group.

Some failure mode groups are ignored for reliability calculation. This type is described in Table 5.22.

Table 5.22 Failure mode group object types

Type	Description
Ignorable Failure Mode Group	The failure mode group is considered highly reliable compared with other items, so it can be ignored when reliability is assessed.

5.4.3.7 Failure Interaction and Dependency Relationships

Some items in a system operate interactively and influence the reliability of one another. This is called ***failure interaction*** and defined as follows:

Definition 5-24

*A **Failure Interaction** is a mutual interaction between two items in a system that causes a system failure.*

Based on this definition, a reliability relationship for failure interactions is represented in mathematical form in Equation (5.70). The failure interaction relationship consists of two reliability objects.

$$\mathbf{rr}_{\mathbf{interaction,i}} \equiv (\mathbf{rrid}, \mathbf{rt}, \mathbf{RRL}_{\mathbf{interaction}}) \tag{5.70}$$

where

rt is a failure interaction relationship type.

$\mathbf{RRL}_{\mathbf{interaction}}$ consists of two elements (e.g. $\mathbf{RRL}_{\mathbf{interaction}}$ = {(heat_source, roid, null), (heat_source, roid, null)}).

In some other cases, the failure of an object depends on the failure of another object. This is called ***failure dependency***, defined as follows:

Definition 5-25

*A **Failure Dependency** is a relation between two items where the reliability of one item is considered to be limited by the reliability of the other item.*

Based on this definition, a reliability relationship for failure dependency is represented in mathematical form in Equation (5.71). The failure dependency relationship consists of two reliability objects.

$$\mathbf{rr}_{\mathbf{dependency,i}} \equiv (\mathbf{rrid}, \mathbf{rt}, \mathbf{RRL}_{\mathbf{dependency}}) \tag{5.71}$$

where

rt is a failure dependency relationship type.

$\mathbf{RRL}_{\mathbf{dependency}}$ consists of two elements (e.g. $\mathbf{RRL}_{\mathbf{dependency}}$ = {(cooling_system, roid, null), (heat_source, roid, null)}).

Electronic systems may fail by electromagnetic interference (EMI) among electrical circuit boards that emit high frequency electromagnetic radiation. Such a system failure is a typical example of failure interaction. Another example is high temperature-induced failure of electronic systems. This may be due to the presence of two or more high power packages assembled in close proximity to each other.

As for failure dependency, a fan for cooling microprocessor CPU and a microprocessor is a typical example, because the reliability of the microprocessor is limited by the reliability of the fan.

5.5 ROM-LIBRARY

Since DF, RPM, and UC belong to the specific system design domain, we select the PWBA design domain as an example specific domain and represent DF, RPM, and UC using UML Class Diagrams in following subsections.

5.5.1 Design Features

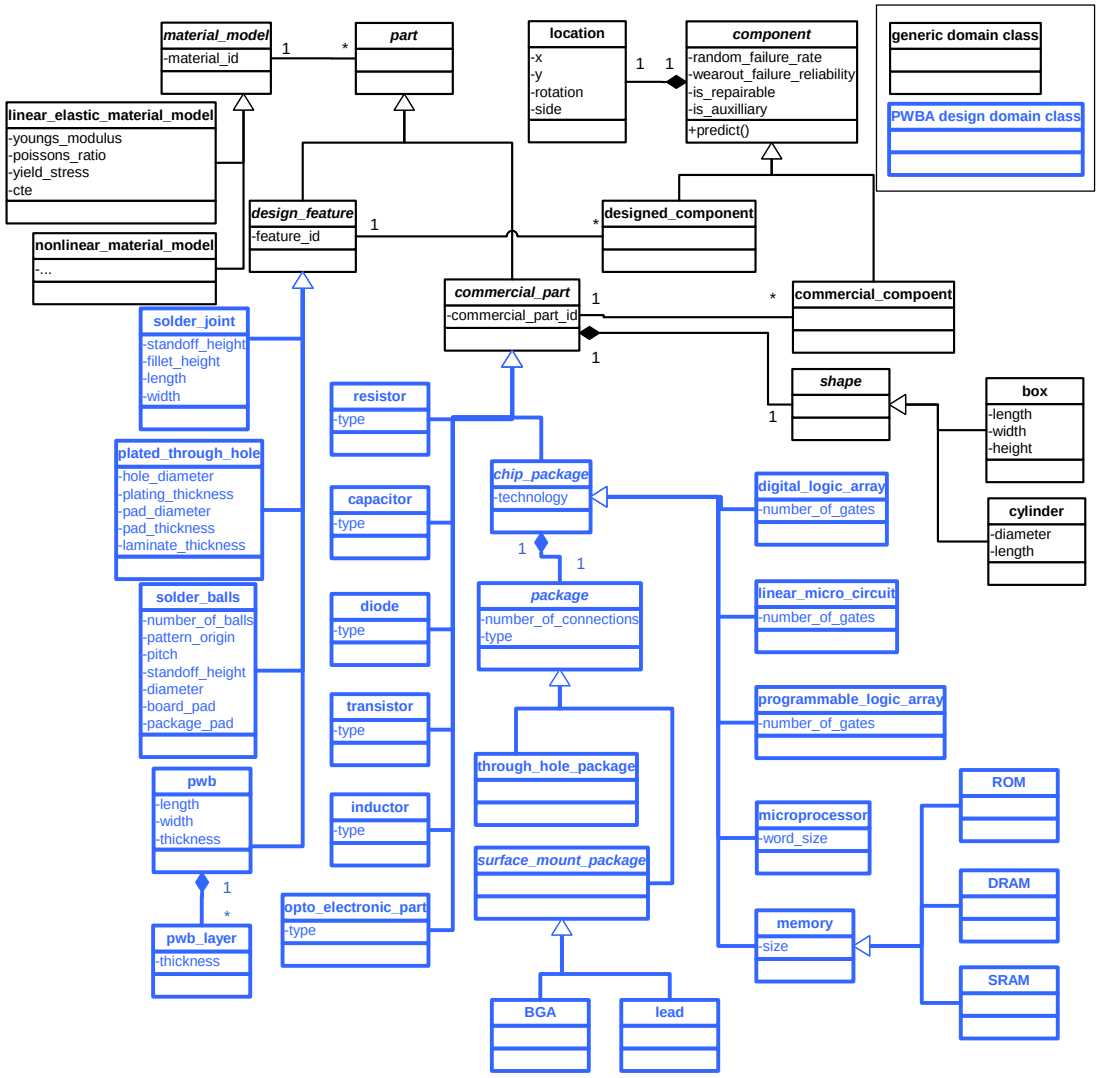


Figure 5.14 ROM-Library Model for design features in the PWBA design domain

For reliability prediction and design recommendations, representing Design Features (DF) is necessary. The UML model for design features in Figure 5.14 includes thirteen leaf type commercial parts (e.g. *resistor*, *capacitor*, *inductor*, *programmable logic array*, *microprocessor*, and so on) and four design features (i.e. *board*, *solder joints*, *plated through hole*, and *solder balls*). These are sub classes of the *part* class that has an association relationship with the *component* class (Figure 5.14). More commercial parts and design features may be added depending on the scope of the design. Similarly, if design features are represented in the mechanical power train assembly domain, the *commercial part* class may be specialized into a *gear* class, a *bearing* class, a *belt* class, and so on.

5.5.2 Reliability Prediction Models

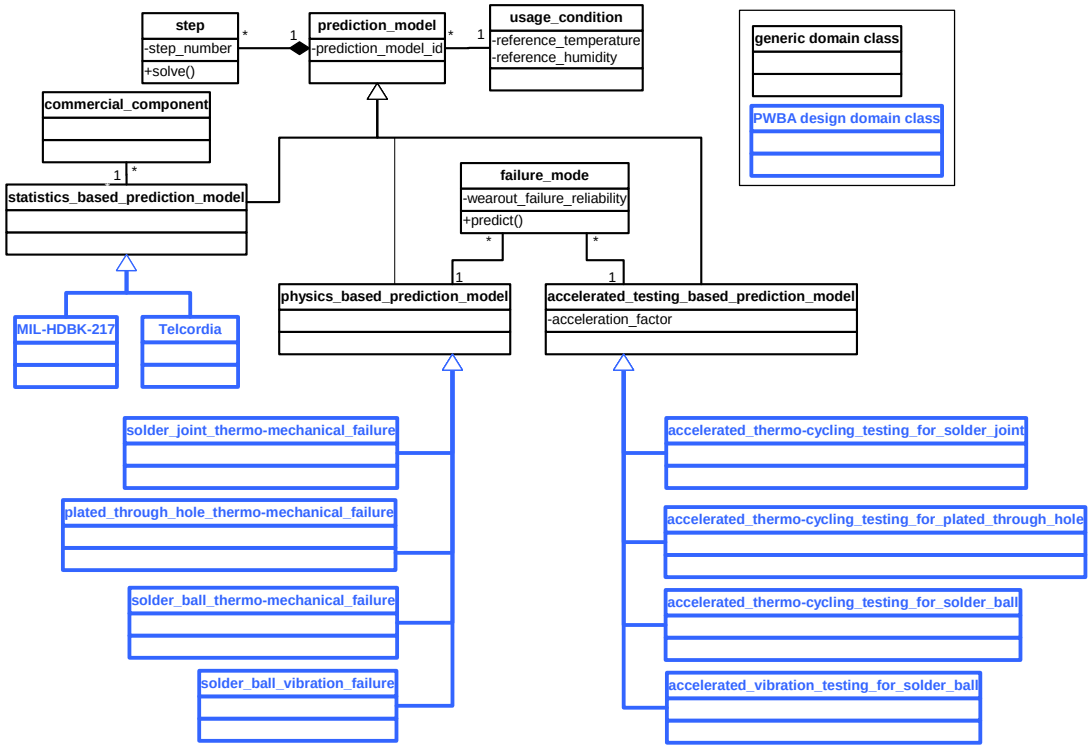


Figure 5.15 ROM-Library Model for reliability prediction models in the PWBA design domain

Reliability prediction models (RPM) are represented for the PWBA design domain in Figure 5.15. The RPM representation starts from the *prediction model* class that consists of multiple steps and one associated usage condition. The *prediction model* class is specialized into three sub classes: *statistics-based prediction model*, *accelerated testing-based prediction model*, and *physics-based prediction model*. The *statistics-based prediction model* is specialized into two sub classes in the PWBA design domain (e.g. *MIL-HDBK-217* and *Telcordia*); the *accelerated testing-based prediction model* is specialized into four sub classes in the PWBA design domain (e.g. *accelerated thermo-cycling testing for solder joint*, *accelerated thermo-cycling testing for plated through hole*,

accelerated thermo-cycling testing for solder ball, and *accelerated vibration testing for solder ball*); and the *physics-based prediction model* is specialized into four sub classes in the PWBA design domain (e.g. *plated through hole thermo-mechanical failure* and *solder joints thermo-mechanical failure* prediction models). Each prediction model class has associated commercial components or failure modes. If RPM is represented in the mechanical power train assembly domain, the *physics-based prediction model* class may be specialized into a *gear wearout failure* class, a *bearing wearout failure* class, and so on.

5.5.3 Usage Conditions

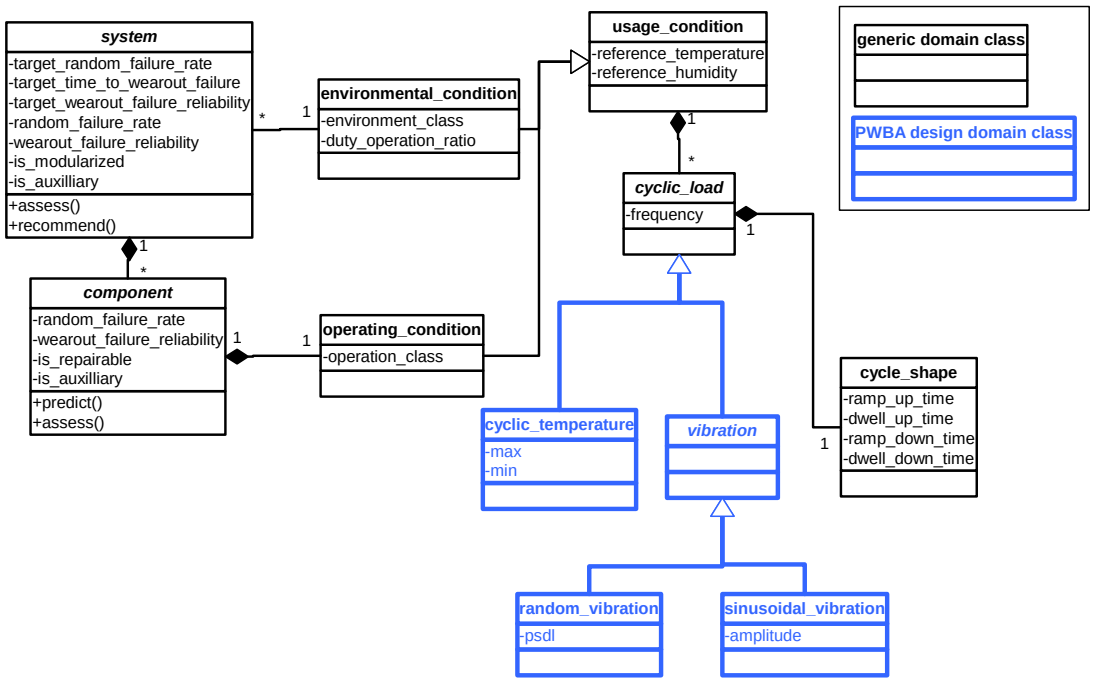


Figure 5.16 ROM-Library Model for usage conditions in the PWBA design domain

Usage conditions (UC) are represented for the PWBA design domain in Figure 5.16. First, the *usage condition* class is defined with two typical properties (i.e. *reference temperature* and *reference humidity*) and multiple cyclic loads (e.g. *cyclic temperature* and *vibration*). It is specialized into two sub classes: *environmental condition* and *operating condition*. The environmental condition affects evenly all the components in the system, while the operating condition may be different from component to component. These relationships are modeled by the association relationship between the *system* class and the *environmental condition* class and the composition relationship between the *component* class and the *operating condition* class. The *cyclic load* is specialized into three sub classes in the PWBA design domain (i.e. *cyclic temperature*, *random vibration*, and *sinusoidal vibration*). Likewise, if usage conditions are represented in the mechanical power train assembly domain, the *cyclic load* class may be specialized into a *cyclic force* class, a *cyclic moment* class, and so on.

5.6 ROM-TREE: A GRAPHICAL VIEW OF SDFR STRUCTURE

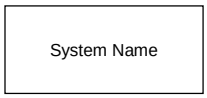
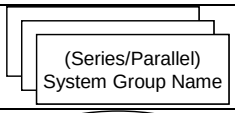
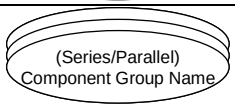
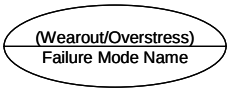
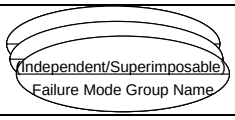
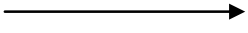
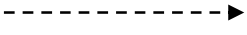
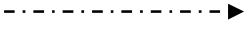
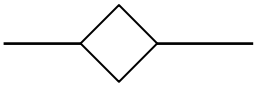
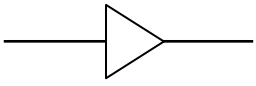
ROM Classes and Relationships in ROM-Structure		ROM-Tree Symbols	
Non-homogeneous System [a,b,c]		Double Line	
Complex Homogeneous System [a,b,c]		Single Thick Line	
Simple Homogeneous System [a,b,c]		Single Regular Line	
Series System Group [a,b,c]		(Series)	
Parallel System Group [a,b,c]		(Parallel)	
Commercial Component [a,b,c]		Single Thick Line	
Designed Component [a,b,c]		Single Regular Line	
Series Component Group [a,b,c]		(Series)	
Parallel Component Group [a,b,c]		(Parallel)	
Wearout Failure Mode [c]		(Wearout)	
Overstress Failure Mode [c]		(Overstress)	
Independent Failure Mode Group [c]		(Independent)	
Superimposable Failure Mode Group [c]		(Superimposable)	
Composition	Static (No Activities)	Single Regular Line	
	Allocation Activities	Single Regular Line with Arrow	
	Assessment Activities	Single Dashed Line with Arrow	
	Prediction Activities	Single Dash-Dot Line with Arrow	
Failure Interaction			
Failure Dependency			
Options: [a] Dashed Line: Modularized or repairable items, [b] Dash-Dot Line: Auxiliary items, [c] Gray-Filled: Ignorable items.			

Figure 5.17 ROM-Tree symbols and options for ROM-Structure

The schema presented in the previous sections is for computer interpretable and processing. For easier human understanding, an equivalent set of graphical symbols and options will be used to represent a ROM instance tree, which is called a Reliability Object Model Tree (ROM-Tree). The ROM-Tree symbols and options are illustrated in Figure 5.17.

In Figure 5.17, a system is graphically represented as a rectangle, and a system group is represented as stacked rectangles. A component is represented as an oval, and a component group is represented as stacked ovals. A failure mode is represented as an oval with cross line, and a failure mode group is represented as stacked ovals with cross line. A reliability allocation activity is represented as a line with arrow, a reliability prediction activity is represented as a dashed line with arrow, and a reliability assessment activity is represented as a dash-dot line with arrow. A failure interaction is represented as a line with diamond, and a failure dependency is represented as a line with triangle.

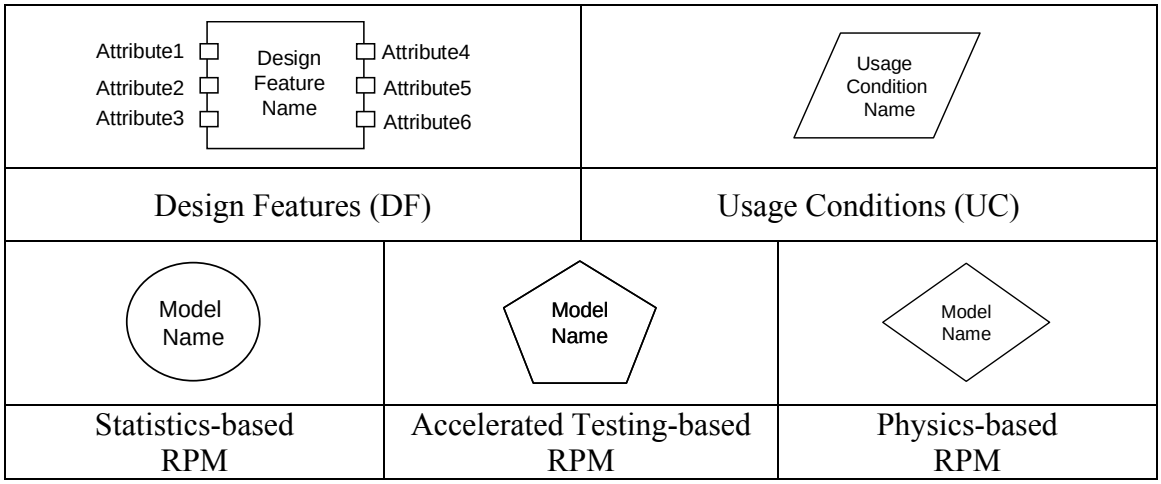


Figure 5.18 ROM-Tree symbols for ROM-Library

ROM-Tree also represents reliability design features (DF), prediction models (RPM), and usage conditions (UC). Design features are represented as a large rectangular box with small rectangular boxes around it for each of its attributes, and usage conditions are represented as parallelograms. Three reliability prediction model types are represented as a circle for a statistics-based RPM, a pentagon for an accelerated testing-based RPM, and a diamond for a physics-based RPM (Figure 5.18). Example ROM-Tree models are given in Chapter 7.

5.7 SUMMARY AND DISCUSSION

This chapter explains Reliability Object Model (ROM). ROM consists of 1) reliability metrics that consider both random failures and wearout failures, 2) algorithms that allocate, predict, and assess reliability, 3) rules for design change recommendations, 4) a new reliability structure, and 5) a graphical representation for SDfR structure called ROM-Tree.

ROM covers concepts ranging from system structure to failure modes, from random failures to wearout failures, and from reliability allocation to design recommendations. The complete support of SDfR by ROM is shown in **Table 5.23**, which presents a comparison between the existing reliability analysis methods and ROM with respect to the SDfR specifications.

As for SDfR specifications, FMEA supports only one level *Assembly Structure*, *Failure Mode Structure*, *Rules for Design Change*, and *Reliability Metrics*. However, it cannot support *Design Features*, *Failure Interactions and Dependencies*, *Logical Structure*, *Reliability Prediction Models*, and *Usage Conditions*. RBD supports *Assembly Structure*, *Logical Structure*, and *Reliability Metrics*. However, it cannot support *Design Features*, *Failure Interactions and Dependencies*, *Failure Mode Structure*, *Rules for Design Change*, *Reliability Prediction Models*, and *Usage Conditions*. FTA supports *Assembly Structure*, *Logical Structure*, and *Reliability Metrics*. However, it cannot support *Design Features*, *Failure Interactions and Dependencies*, *Failure Mode Structure*, *Rules for Design Change*, *Reliability Prediction Models*, and *Usage Conditions*. Markov Chain supports only *Failure Interactions and Dependencies* and *Reliability Metrics*.

As for SDfR activities, existing methods support reliability assessment fully and design recommendation partially. However, they don't support reliability allocation and prediction. Therefore, they are limited to support the top-down and bottom-up approach of SDfR.

ROM supports all the SDfR specifications and activities. Therefore, we think that ROM provides a complete framework for SDfR compared with existing approaches including richer semantics and unified methods.

Table 5.23 A comparison between existing reliability analysis methods and ROM

SDfR Specifications and Activities	RBD	FTA	FMEA	Markov Chains	ROM
SDfR Specifications					
Assembly Structure	○	○	○		●
Design Features					●
Failure Interactions and Dependencies				○	●
Failure Mode Structure			○		●
Logical Structure	●	●			●
Rules for Design Change Recommendations			○		●
Reliability Metrics	●	●	○	●	●
Reliability Prediction Models					●
Usage Conditions			○		●
SDfR Activities					
Reliability Allocation					●
Reliability Prediction					●
Reliability Assessment	●	●	●	●	●
Design Recommendation		○	○		●

●: Strongly Supported, ○: Partially Supported, Blank: Not Supported.

ROM inherits the benefits of object-oriented schemes. For example, ROM integrates reliability analysis structure, reliability metrics and reliability activities in a consistent and explicit manner using relationships, properties, and method definitions of

object-oriented schemes. In addition, modularity and extensibility are applied to support multi-disciplinary and multi-level aspects of SDfR. Finally, the encapsulation characteristic of object-oriented schemes makes the SDfR algorithms simple and efficient. Therefore, we conclude that ROM represents reliability knowledge in a more efficient manner than existing methods.

ROM implementation is described in Chapter 6, and the use of ROM are presented in Chapter 7 with four electronic system test cases.

CHAPTER 6

RELIABILITY OBJECT MODEL (ROM) IMPLEMENTATION

This chapter describes an information framework and developed prototype CASDfR tools that implement reliability object model (ROM).

6.1 INTRODUCTION TO INFORMATION FRAMEWORK

A framework is an abstract structure for describing complex concepts, processes, or methods [Webster's College Dictionary, 1997]. An information framework is a logical structure of information flow for application development or system integration. Such an information framework is essential for collaborative design, and diverse frameworks have been developed for various problems in engineering collaboration.

Engineering frameworks can be classified by applied information technologies, application domains, and integration strategies. First, the classification of engineering frameworks by applied information technologies includes frameworks based on database [Rosenman and Gero, 1996], web-based frameworks [Rezayat, 2000], and standards-based framework [Bajaj et al., 2003; Pratt et al., 2005]. Second, the classification of engineering frameworks by application domains includes design-for-manufacturability (DfM) frameworks [Bajaj et al., 2003; Zhao and Shah, 2005], modeling and simulation frameworks [Shellgren and Drogou, 1998; Peak et al., 1998], product data management (PDM) frameworks [Fenves et al., 2003], and product lifecycle management (PLM)

frameworks [Sudarsan et al., 2005]. Third, the classification of engineering frameworks by integration strategies includes model-based frameworks [Peak et al., 1998], Application Protocol Interface (API)-based frameworks [Penoyer et al., 2000], meta-information based frameworks [Cho et al., 2006], and generic domain shell-based frameworks [Zhao and Shah, 2005].

As an information framework for SDfR domain, we've developed CASDfR-Framework. We define CASDfR-Framework as follows:

CASDfR-Framework *is an information framework for SDfR.*

We adapt the idea of Multi-Representation Architecture (MRA) [Peak et al., 1998] for the integration of diverse reliability engineering tools in CASDfR-Framework. MRA is a framework for CAD and CAE integration. Peak argued that the gap between design and analysis models is too large for a single general integration bridge, and therefore divides the MRA into four information representations that act as stepping stones between the design and analysis tool extremes [Peak et al., 1998]. These four information representations are solution method models (SMMs), analysis building blocks (ABBs), analyzable product models (APMs), and context-based analysis models (CBAMs). These models are illustrated in Figure 6.1 [Peak et al., 1998; Tamburini, 1999].

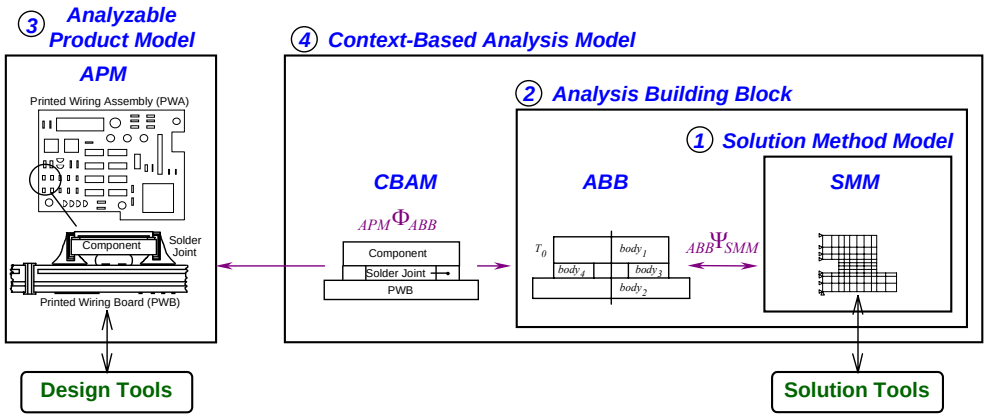


Figure 6.1 Multi-Representation Architecture for design-analysis integration

MRA is a model-based integration strategy, and an alternative way is a function-based integration strategy using Application Protocol Interface (API) [Penoyer et al., 2000]. Since an API depends on a specific tool, the function-based integration strategy is not flexible, while the model-based integration strategy is flexible because any tool can be integrated with tool-independent model formats like STEP APs [Kemmerer, 1999].

The development of CASDfR-Framework starts with a use case scenario for SDfR. Through this scenario, diverse engineering domains and models for SDfR are identified. The generalization of this layout leads to CASDfR-Framework. In the following sections, we describe a use case scenario for SDfR, CASDfR-Framework, and CASDfR tools that implement ROM. These are illustrated in Figure 6.2.

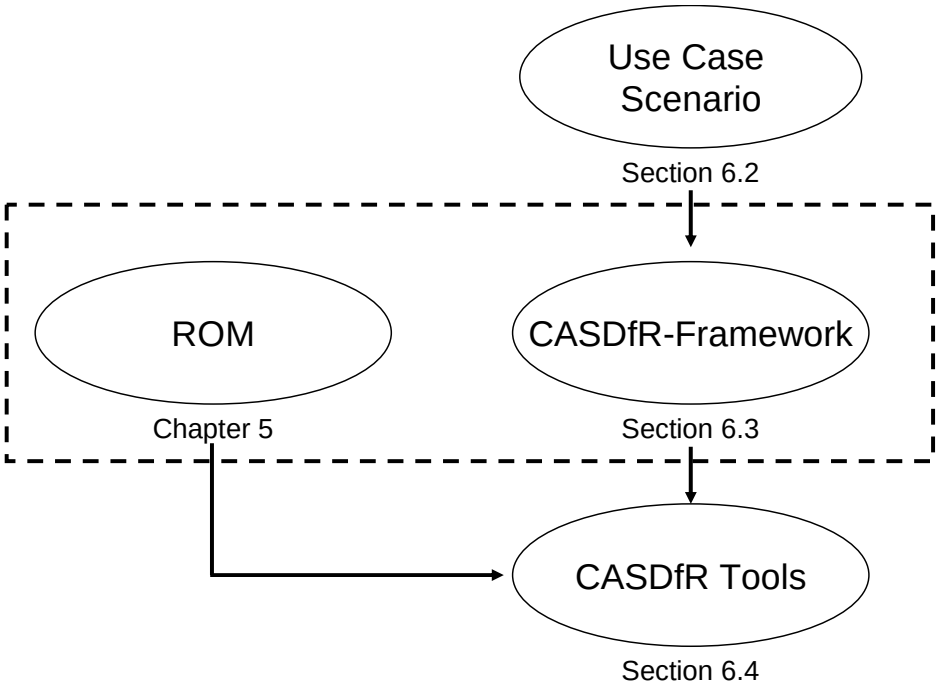


Figure 6.2 Overview of Chapter 6

6.2 A USE CASE SCENARIO FOR SDFR

For SDfR, multiple groups are inevitably involved in system design because of its multi-level and multi-disciplinary nature. To manage multiple groups efficiently, a system design manager must coordinate multi-level and multi-disciplinary engineering groups. The system design manager start system design, and then he or she transfers the system configuration information to a system reliability engineer and subsystem designers (①, ② in Figure 6.3). The system reliability engineer sets up and allocates target reliability. Then, subsystem designers begin subsystem design. The target subsystem reliability and the subsystem design information are transferred to subsystem reliability engineers (③, ④ in Figure 6.3).

Then, they predict component reliability with the cooperation of simulation engineers, and assess subsystem reliability based on their level or disciplinary knowledge (⑤, ⑥ in Figure 6.3). If assessed subsystem reliability is less than the allocated target reliability, subsystem designers and reliability engineers modify the subsystem designs (⑦ in Figure 6.3). Then, the subsystem reliability engineers update the assessed subsystem reliability to the system reliability engineer (⑧ in Figure 6.3), and the subsystem designers update the design changes to the system design manager (⑨ in Figure 6.3). Then, the system reliability engineer assesses system reliability and modifies the system design if necessary (⑩ in Figure 6.3).

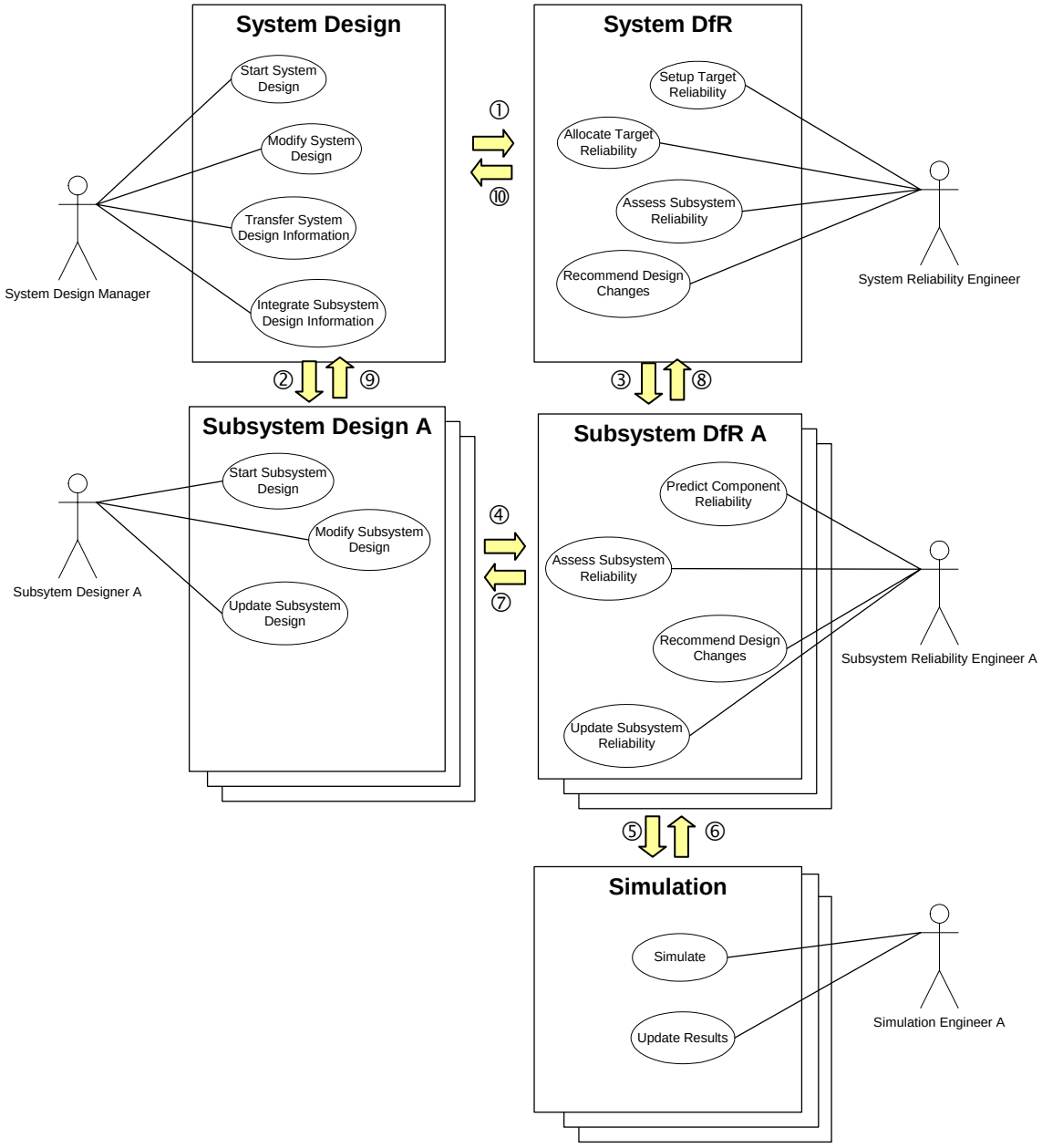


Figure 6.3 A use case scenario of SDfR

6.3 CASDFR-FRAMEWORK

From this use case scenario, five distinct domains are identified: the generic design domain, specific design domains, the generic SDfR domain, specific SDfR domains, and specific simulation domains.

In the generic design domain, a system design manager starts and completes system design. He or she starts system design developing system requirement and configuration models. He or she completes system design integrating and qualifying embodied subsystem models.

In specific design domains, subsystem designers embody subsystems based on the subsystem requirement models. They select commercial parts and design interconnection components to realize subsystem models.

In the generic SDfR domain, a system reliability engineer creates a ROM-Structure model according to the system configuration model developed by the system design manager. The system reliability engineer sets up target reliability and allocates the target reliability according to the developed ROM- Structure model. After subsystem designs and subsystem reliability assessments are completed, the system reliability engineer integrates and qualifies subsystem reliabilities. If design changes are required, they recommend design alternatives to the system design manager.

In specific SDfR domains, subsystem reliability engineers create subsystem ROM-Structure based on subsystem design models and ROM-Library models based on specific reliability domain knowledge. They predict component reliabilities and assess subsystem reliabilities according to the developed ROM models. If the assessed

reliabilities of the subsystems are less than the allocated target reliabilities, they recommend design alternatives to the associated subsystem designers.

In simulation domains, simulation engineers create simulation models and execute requested simulations for component reliability prediction. They return the simulation results to the associated subsystem reliability engineers.

Such engineering models capture their own domain information and knowledge for system design, and they are processed across different domains by manual or automation. For example, a ROM-Structure model in the generic SDfR domain is downloaded to specific SDfR domains. After this ROM-Structure model is linked to ROM-Library model, component simulation information is downloaded to specific simulation domains. After each simulation is finished, the results are uploaded to the associated specific SDfR model. Similarly, after each subsystem SDfR is finished, the results are uploaded to the generic SDfR model.

These model operations and associations are illustrated in Figure 6.4. The white box arrows and the numbers in circles represent model operations, and the number of rectangles represents cardinalities between two different domains and models.

The structure in Figure 6.4 is called CASDfR-Framework. This framework shows the information flow of SDfR across multiple domains and provides a base to develop and integrate various engineering S/W tools. Such a tool integration capability of the framework helps to meet multi-disciplinary support requirements of SDfR.

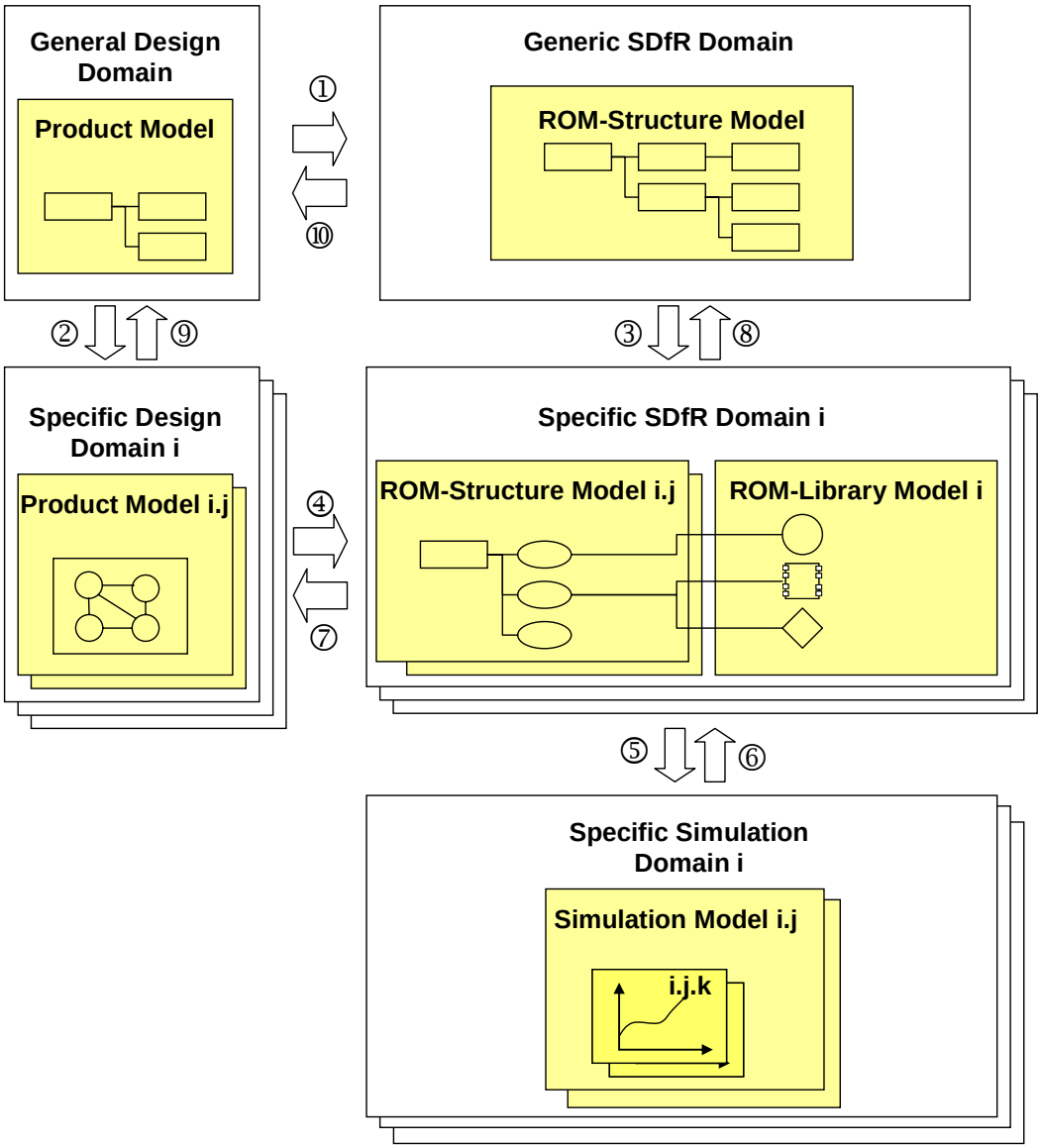


Figure 6.4 CASDfR-Framework

6.4 PROTOTYPE CASDFR TOOLS

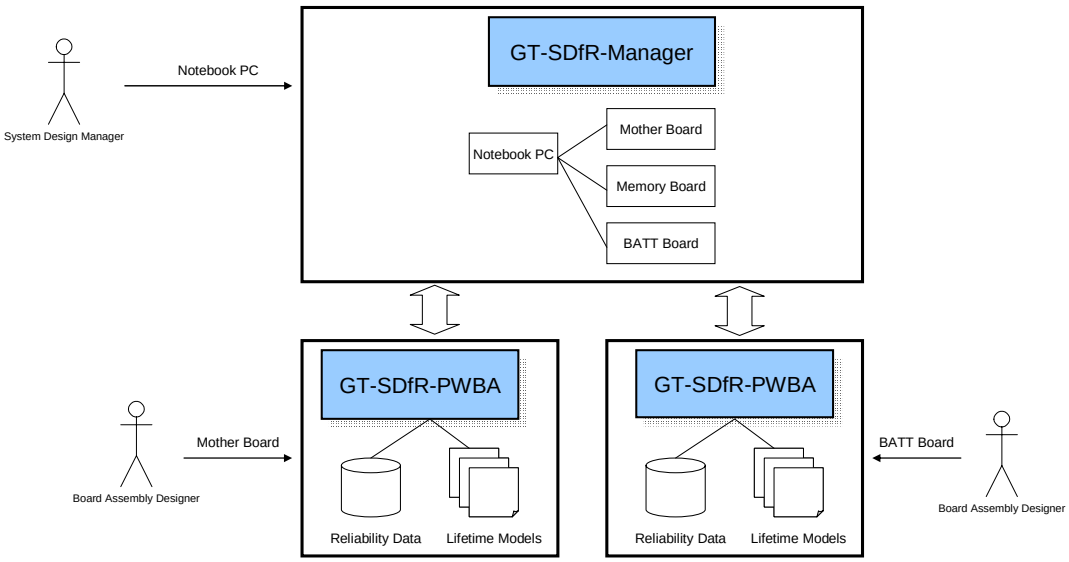


Figure 6.5 CASDfR tools for notebook PC design

According to CASDfR-Framework, one generic SDfR tool and one specific SDfR tool are required for notebook PC design (Figure 6.5). The generic SDfR tool, called GT-SDfR-Manager, is used for allocating target reliability, and assessing system-level reliability. The specific SDfR tool, called GT-SDfR-PWBA, is used for predicting component reliability, and assessing electronic board-level reliability.

For the development of such CASDfR tools in a systematic and consistent manner, we've followed the three-level architecture, which is developed for systematic and consistent database design and implementation [Elmasri and Navathe, 1994]. The three-level architecture consists of conceptual level, logical level, and physical level. In the conceptual level of the architecture, the semantics of universe-of-disclosure are modeled, and the model is for human interpretation. The semantic model developed in the conceptual level is converted to a tool-independent model for implementation in the

logical level. Finally, the logical level model is implemented in a specific tool in the physical level.

For the CASDfR tool development, the conceptual level model that captures SDfR knowledge is ROM. The logical level model that describes tool-independent, implemental information is an EXPRESS [Schenck and Wilson, 1994] version of ROM[15]. The physical level model that consists of executable scripts or codes is ROM implementation codes in JAVA™ [SUN, 2006] and XML [W3C, 2003], which are the LKSoft's STEP Book [LKSoft, 2006] platform. These are illustrated in Figure 6.6.

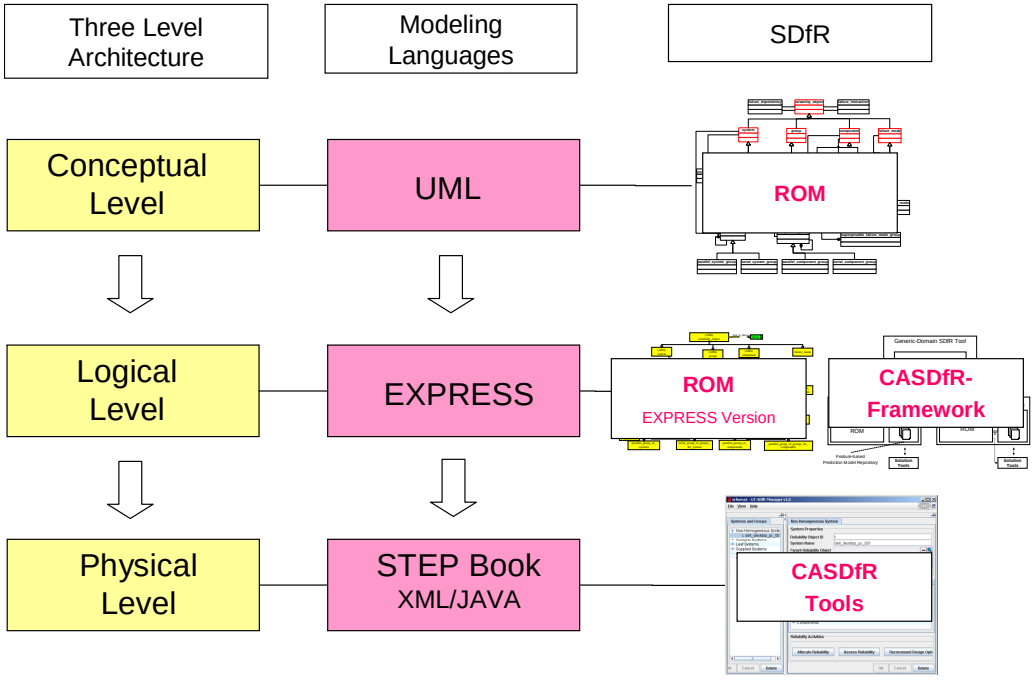


Figure 6.6 Three-level architecture for the CASDfR tool development

Through this three-level architecture, GT-SDfR-Manager (Figure 6.7) and GT-SDfR-PWBA (Figure 6.8) are developed. With GT-SDfR-Manager, system reliability engineers start system design for reliability, set up target reliability, allocate target reliability, and assign subsystem target reliability. After updating subsystem reliability,

[15] The EXPRESS version of ROM is shown in Appendix D.

the system reliability engineers also assess system reliability and recommend system design alternatives if necessary.

Figure 6.7 illustrates ROM implementation in GT-SDfR-Manager. ROM-Structure is implemented as a tree browser. ROM metrics are implemented as an array of input boxes. ROM activities are implemented as an array of buttons. For example, Figure 6.7 illustrates reliability objects of *Video Encoder System 001* implemented in GT-SDfR-Manager. Systems, system groups, and a failure dependency are shown in the left side tree browser. Target reliability metrics of *Video Encoder System 001* are shown in the middle of the right side tab. Three reliability activities of *Video Encoder System 001* are shown in the bottom of the right side tab. The detail of the video encoder system test case is described in Chapter 7.

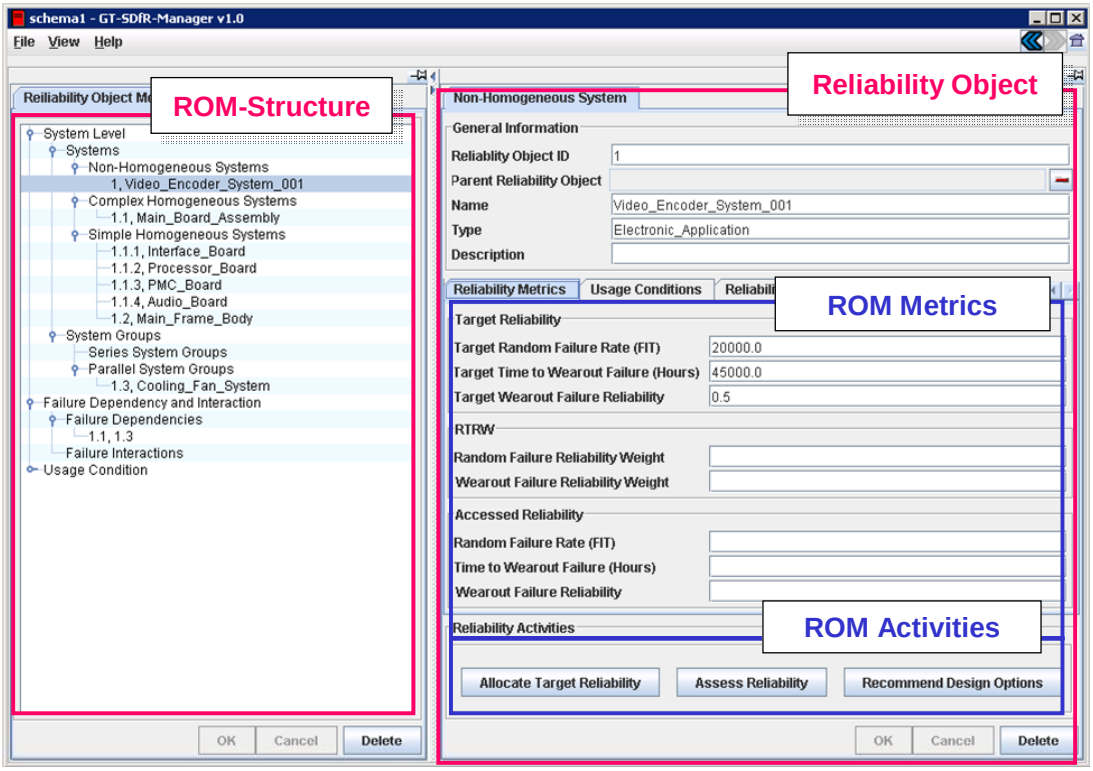


Figure 6.7 GT-SDfR Manager

With GT-SDfR-PWBA, subsystem reliability engineers predict component reliability, and assess subsystem reliability. If assessed subsystem reliability is less than allocated target reliability, the subsystem reliability engineers recommend component design changes. Then, they update subsystem reliability to system reliability engineers.

Figure 6.8 illustrates reliability objects of a USB board assembly in GT-SDfR-PWBA. ROM-Structure and ROM-Library are implemented as a tree browser. ROM metrics and ROM activities are implemented as the same way with GT-SDfR-Manager.

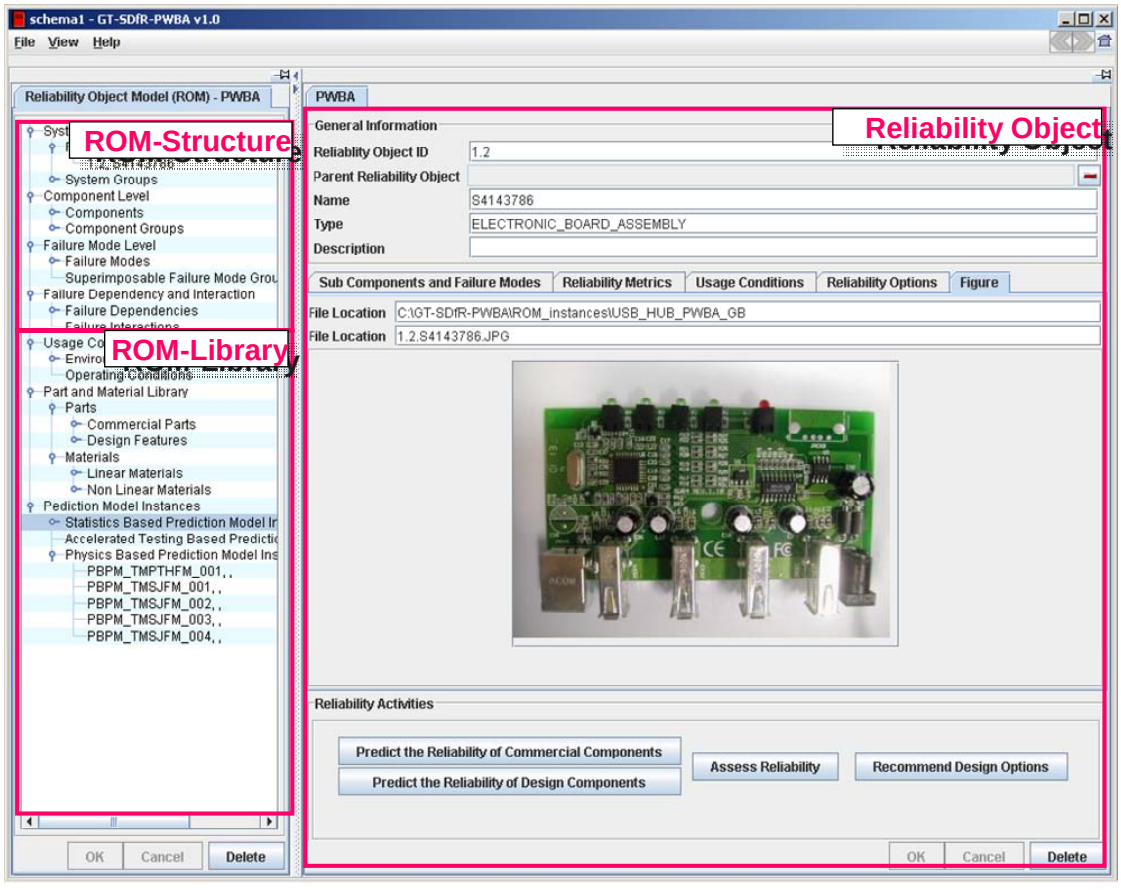


Figure 6.8 GT-SDfR-PWBA

GT-SDfR-PWBA also integrates diverse simulation tools such as ANSYS™ [ANSYS, 2005], Matlab™ [Mathworks, 2005], and Oracle™ [Oracle, 2006] (Figure 6.9). These tools execute reliability prediction models generated by GT-SDfR-PWBA, and update the prediction results to GT-SDfR-PWBA. For example, APDL files [ANSYS, 2005] that are ANSYS script files are generated by GT-SDfR-PWBA and executed by ANSYS. The simulation results are updated to GT-SDfR-PWBA by automation and used to calculate associated component reliability.

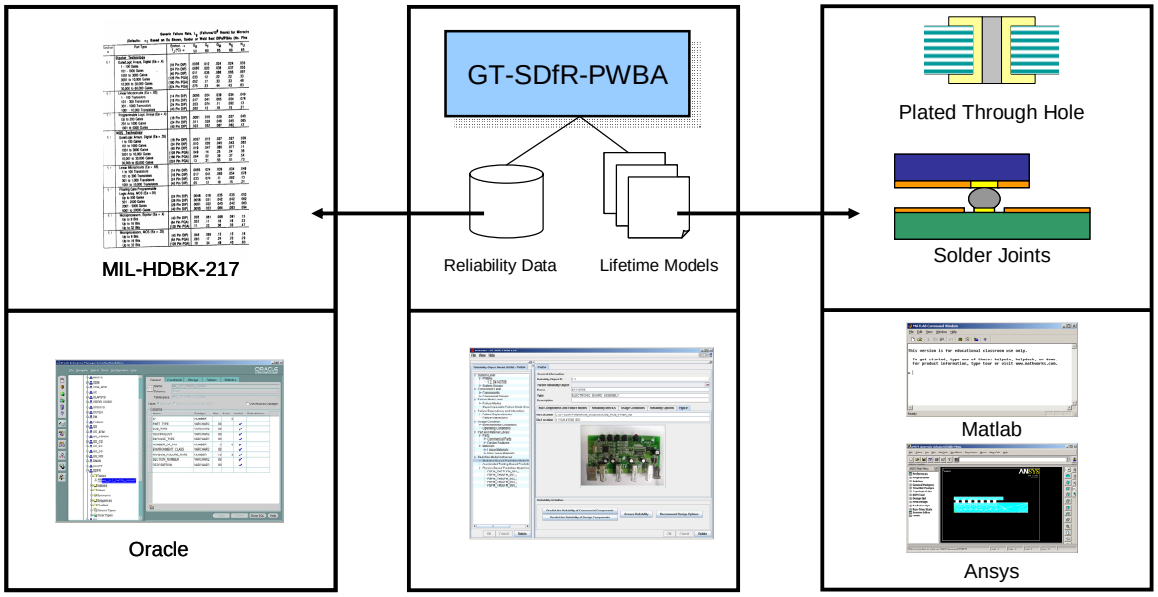


Figure 6.9 Integration of simulation tools in GT-SDfR-PWBA

6.5 SUMMARY AND DISCUSSION

This chapter explains CASDfR-Framework and introduces developed prototype CASDfR tools. CASDfR-Framework is an organized structure of multi-domain models for collaborative system design among various engineering groups. If we classify this framework, it is a framework based on an object-oriented database (LKSoft's STEP Book), a framework for SDfR domain, and a model-based framework. It can also incorporates standards like STEP APs to achieve flexibility by supporting diverse engineering models and tools.

According to CASDfR-Framework, the prototype CASDfR tools that implement reliability object model (ROM) are developed. The prototype CASDfR tools increase process efficiency by automating model operations and system design quality by reducing human errors. The implementation results show that ROM is computationally efficient.

CHAPTER 7

TEST CASES

In this chapter, ROM is demonstrated by four electronic systems: a video broadcasting system demonstrates the target random failure reliability allocation (section 7.1); a notebook PC demonstrates the target wearout failure reliability allocation (section 7.2); a USB hub demonstrates the reliability prediction and assessment (section 7.3); and a mockup Engine Control Unit (ECU) demonstrates the design recommendations (section 7.4).

7.1 ROM-BASED TARGET RELIABILITY ALLOCATION FOR VIDEO BROADCASTING SYSTEM DESIGN

7.1.1 Overview of a Video Broadcasting System Test Case

Video encoders compress video signals for efficient transmission in broadcasting service. Since multi-channel signals are broadcast at the same time, multiple video encoders are required. In most cases, redundant video encoders are used for reliable video broadcasting service without any interruption. Figure 7.1 illustrates six encoders for six channels and two redundant encoders for backups (4th Level). Each video encoder (3rd Level) consists of electronic board systems, and each board system (2nd

Level) consists of various electronic packages (1st Level). These are illustrated in Figure 7.1.

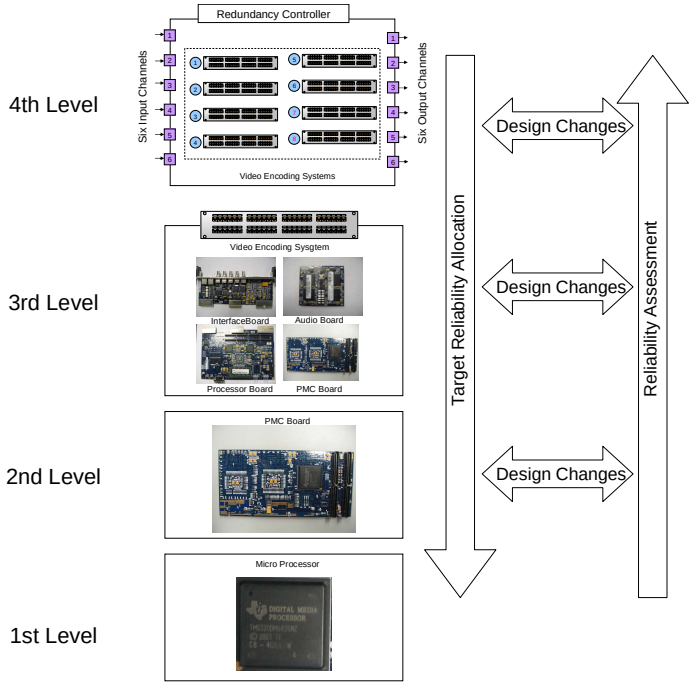


Figure 7.1 A video broadcasting system[16]

Since video broadcasting systems are working in well conditioned environments, random failures of encoder systems are more dominant and important than wearout failures. Therefore, the design of a reliable video broadcasting system against random failures is considered using the ROM method. For example, the target reliability of the video broadcasting system is allocated from the 4th level to the 2nd level. Then, the reliability of board systems is predicted. Depending on the difference between the allocated target reliability and the assessed reliability, various design changes are

[16] The Video Broadcasting Systemin in Figure 7.1 is a product of EGT Inc. (www.egtinc.com), and EGT Inc. permits the release of its figures and information for academic purpose only (February 2007).

considered to satisfy the target reliability of the video broadcasting system illustrated in Figure 7.1.

Rationales for selecting this test case are summarized as follows:

- Random failures of video broadcasting systems are issues in broadcasting service.
- Video broadcasting systems consist of multi-level assembly structures.
- Video broadcasting systems consist of both parallel and series structures.
- Design information of video broadcasting systems is available from EGT Inc.

7.1.2 ROM-Tree Model for a Representative Video Broadcasting System

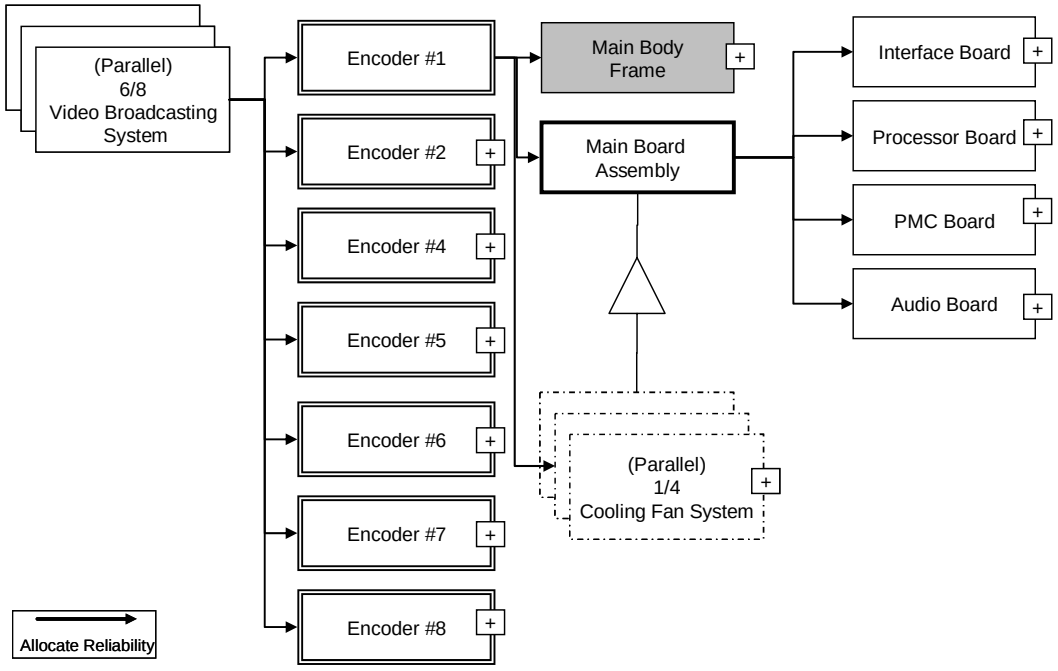


Figure 7.2 A ROM-Tree model for the EGT video broadcasting system[17]

[17] Figure 7.2 illustrates the simplified structure of EGT product according to the agreement with EGT Inc.

Figure 7.2 shows a ROM-Tree model for a video broadcasting system. This model starts from the top level system, *Video Broadcasting System*, which consists of eight identical encoder systems. Since six encoders are for six channels and two redundant encoders are for backups, *Video Broadcasting System* is a 6-out-of-8 parallel system group. This system group is represented using stacked rectangles.

Each *Encoder* consists of *Main Body Frame*, *Main Board Assembly*, and *Cooling Fan System*. Since an *Encoder* includes mechanical and electrical subsystems together, it is represented with a double-line box indicating a non-homogenous system per the ROM-Tree notation given in Figure 5.17.

The gray-filled box named *Main Body Frame* indicates it is being modeled as an ignorable system with respect to reliability. Our ROM method does not include such systems in target reliability allocation because they are considered either irrelevant to the main functionalities of the top level system or highly reliable compared with other systems.

The dot-dashed box of *Cooling Fan System* indicates it is modeled as an auxiliary system group, which is not directly related to the main functionalities of the top level system but important for the reliability of other systems. Our ROM method does not include auxiliary systems in target reliability allocation. However, the target reliability of the auxiliary system is typically related to the target reliability of associated systems. For example, the reliability of *Cooling Fan System* must be greater than the reliability of *Main Board Assembly* it is cooling, which is represented by the failure dependency blank triangle in Figure 7.2.

The single thick line box of *Main Board Assembly* indicates it is modeled as a complex homogenous system, which consists of four simple homogeneous systems: *Interface Board*, *Processor Board*, *PMC Board*, and *Audio Board*.

7.1.3 Target Random Failure Reliability Allocation

Following the ROM-Tree structure described in the previous section, the target random failure reliability of the video broadcasting system is allocated to its subsystems. This section explains target random failure reliability allocation for parallel structures from the video broadcasting system to encoder systems (from the 4th level to the 3rd level). Then, it explains the target random failure reliability allocation for series structures from an encoder system to electronic board systems (from the 3rd level to the 2nd level).

7.1.3.1 Target random failure reliability allocation to encoder systems

The video broadcasting system in Figure 7.1 consists of a 6-out-of-8 parallel structure with identical encoders, which means that more than six encoders should work out of eight encoders. Therefore, we use the algorithm presented in the Section 5.3.3.2.

First, we set the target reliability of the video broadcasting system for random failures as follows:

$\lambda_{\text{target, video broadcasting system}}$ =5000 FIT (Failures-in-Time, failures/10^9 hours)

The value of 5000 FIT target random failure rate means that five failures may occur out of 100 identical video broadcasting systems in a year approximately.

Second, we set RTRWs for encoders. Since eight encoders are identical, we use the uniform RTRW ($RTRW^u$) in Equation (5.23) as follows:

$RTRW^u_{i.j,\ encoder} = 1/8$

Third, we follow the steps in the Section 5.3.3.2 for allocating target random failure reliability for parallel structures:

STEP 1: Estimate the initial random failure rate of the minimum-weight subsystem ($\lambda_{i.\min}^{1st}$).

STEP 2: Estimate the random failure rate of the system (λ_i^{1st}).

STEP 3: Calculate the difference between the given target random failure rate of the system (λ_i^G) and the estimated random failure rate of the system (λ_i^{1st}).

STEP 4: Estimate the second time-guess random failure rate of the minimum-weight subsystem ($\lambda_{i.\min}^{2nd}$).

STEP 5: Repeat STEPS 2, 3, and 4 until updated error ratio is smaller than a given tolerance ($ER_\lambda^{n-th} < T_\lambda^G$).

STEP 6: Calculate the allocated target random failure rates of the subsystems ($\lambda_{i.j}^{n-th}$).

The conversancy of normalized error ratio (tolerance 0.001) is illustrated in Figure 7.3.

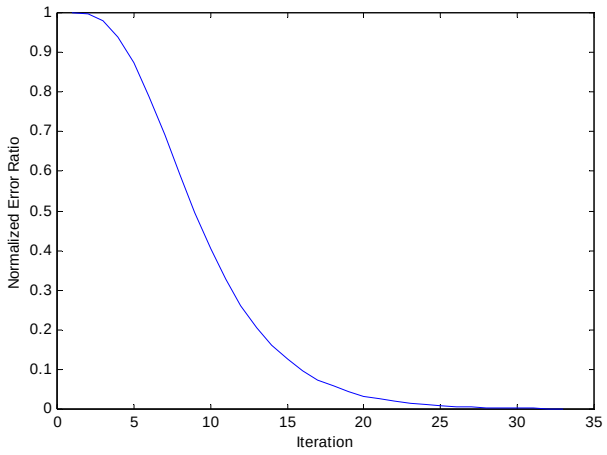


Figure 7.3 Conversancy of normalized error ratio

The allocated target random failure rate of each encoder is as follows:

$\lambda_{\text{target,encoder}}^{n-th}$ = 5782.4 FIT

For checking the validity of this value, we compare the allocated target random failure rates of encoders with different parallel structures. These are shown in Table 7.1. According to the results in Table 7.1, the allocated target random failure rate of the encoder with a 6-out-of-8 parallel structure is between those of the encoders with 5-out-of-8 and 7-out-of-8 parallel structures, and the values are gradually increasing as expected.

Table 7.1 Random failure reliability allocation for video broadcasting system design

Parallel Structure	8-out-of-8	7-out-of-8	6-out-of-8	5-out-of-8	1-out-of-8
Allocated random failure rate (FIT)	625.0	2789.6	5782.4	9553.8	45517.0

7.1.3.2 Target random failure reliability allocation to electronic board systems

Since the target reliability of each encoder system is allocated from the video broadcasting system, the next step is to allocate the target reliability of the encoder system to its series subsystems: *Interface Board*, *Processor Board*, *PMC Board*, and *Audio Board* (Figure 7.2).

First, we set RTRWs for the board systems. Table 7.2 shows $RTRW^d$ calculation results for the board systems. We count the approximate number of components in each board and calculate $ENC_{i.j}$ assuming limited preliminary design information. For $ECSCC_{i.j}$ calculation, we set guide lines for the PWBA domain based on experience: the complexity of BGA chip packages to be 10 and that of a large I/O (>25) chip package to be 5. For example, since the processor board includes six BGA chip packages, the expected complexity sum is 60. After the calculation of $ENC_{i.j}$ and $ECSCC_{i.j}$, we calculate $TENC_i$ that is the sum of all $ENC_{i.j}$ and $TECSCC_i$ that is the sum of all $ECSCC_{i.j}$. From these, we calculate all $RENC_{i.j}$ and $RECSCC_{i.j}$. With the assumption of w_a and w_b to be equal (i.e., $w_a = w_b = 0.5$), we calculate $RTRW_{i.j}$. A high value for $RTRW_{i.j}$ indicates a comparatively less reliable subsystem, and thus a comparatively large target random failure rate.

Per the $RTRW^d$ values shown in Table 7.2, the target random failure reliability of the encoder system is allocated to its series subsystems. For example, the target random failure reliability of the PMC board is calculated by multiplying the target random failure

of the encoder system and its RTRW (Equation (5.25)). The complete results are shown in Figure 7.4.

Table 7.2 RTRWs for the *Encoder* subsystems

Subsystem Name, i.j	**$RENC_{i.j}$**	**$RECSCC_{i.j}$**	**$RTRW_{i.j}$**
Interface Board, i.1	1000/2800 =0.3571	20/165 =0.1212	0.5×0.3571+0.5×0.1212 =0.2392
Processor Board i.2	1100/2800 =0.3929	60/165 =0.3636	0.5×0.3929+0.5×0.3636 =0.3782
PMC Board i.3	500/2800 =0.1786	65/165 =0.3939	0.5×0.1786+0.5×0.3939 =0.2863
Audio Board, i.4	200/2800 =0.0714	20/165 =0.1212	0.5×0. 0714+0.5×0. 1212 =0.0963
• $TENC_i$ = 2800 (= 1000 +1100 + 500 + 200), • $TECSCC_i$ = 165 (= 20 + 60 + 65 + 20).			

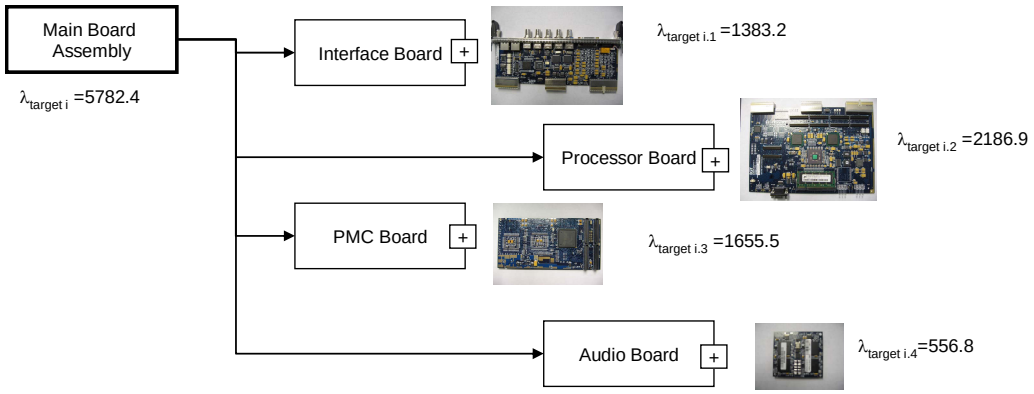


Figure 7.4 Target random failure reliability allocation for the *Encoder* subsystems

If assessed random failure rate of the PMC board is larger than the allocated target random failure rate, design changes are required. The best way is to add redundant components in the PMC board. However, in some cases adding many redundant components makes the PMC board design complex. Therefore, dual PMC boards might

be better practically. Another possible design change may be to add more redundant encoders. For example, a 6-out-of-9 parallel structure is possible instead of the 6-out-of-8 parallel structure. However, this design change may cost more than the design change of adding a redundant PMC board.

7.1.4 Discussion

In the previous sections, the ROM method for target random failure reliability allocation is demonstrated using a video broadcasting system test case. First, the target random failure reliability of the video broadcasting system is allocated to each encoder system using the algorithm for parallel structures. Then, the target random failure reliability of each encoder system is allocated to each electronic board system using the algorithm for series structures.

We have validated the reliability allocation algorithms in Section 5.3.3. In this section, we validate RTRW calculation. While $RTRW^u$ calculation is obvious, $RTRW^d$ calculation is empirical. We set up the $RTRW^d$ calculation guidelines in Section 7.1.3.2. The estimated $RTRW^d$ for the encoder subsystems in Table 7.2 are compared with assessed reliability weights[18], $w_{i.j,assessed}$, based on constant random failure rates provided by EGT Inc. Table 7.3 shows the results of comparison. Since the estimated $RTRW^d$ values are in good agreement with the calculated reliability weights, we think the $RTRW^d$ calculation guidelines are valid for electronic board assembly design.

[18] EGT Inc. engineers designed the encoder subsystems without using any formal allocatiaon algorithm. In this dissertation research, we computed the actual resulting weitghts, $w_{i.j,assessed}$, using the assessed reliabilities for this actual design using Equation (5.21) (where these assessed reliability values came fom historical handbook data).

Table 7.3 Evaluation of the $RTRW^d$ calculation guidelines

Subsystem Name	$RTRW^d$	$w_{i.j,assessed}$	Error Ratio*
i.1 Interface Board	0.2392	0.2518	0.050
i.2 Processor Board	0.3783	0.3974	0.048
i.3 PMC Board	0.2863	0.2739	0.045
i.4 Audio Board	0.0963	0.0768	0.253

* Error Ratio= $| RTRW^d - w_{i.j,assessed}| / w_{i.j,assessed}$.

7.2 ROM-BASED TARGET RELIABILITY ALLOCATION FOR NOTEBOOK PC DESIGN

7.2.1 Overview of a Notebook PC Test Case

Figure 7.5 A representative notebook PC (IBM ™ ThinkPad 390E [IBM, 2006])

A representative notebook PC, IBM™ ThinkPad 390E [IBM, 2006] (Figure 7.5) is chosen for demonstrating the ROM method for target wearout failure reliability allocation. Rationales for selecting this test case are as follows:

- The reliability of notebook PCs has been problematic.
- Most notebook PCs consist of more than 20 subsystems, so they are complex enough to explain the ROM method and target reliability allocation.
- The structure of Notebook PCs is easy to explain.
- Design information of notebook PCs is available from online manuals.

7.2.2 ROM-Tree Model for a Representative Notebook PC

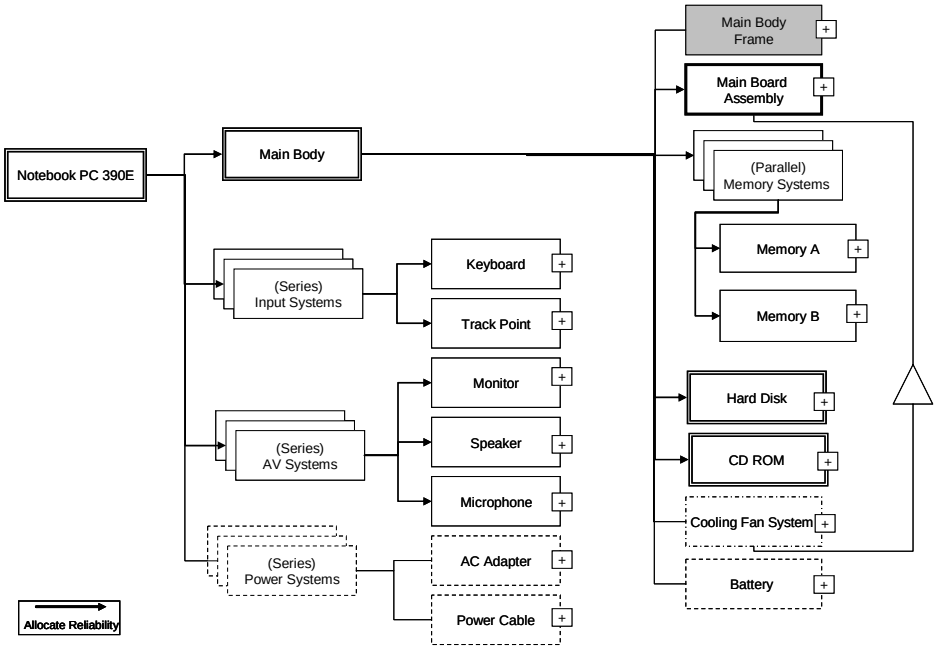


Figure 7.6 A ROM-Tree model for a representative notebook PC[19]

Figure 7.6 shows a ROM-Tree model for a representative notebook PC. This model starts from the top level system, *Notebook PC 390E*, which includes mechanical and electrical subsystems together. Therefore, it is represented with a double-line box indicating a non-homogenous system per Figure 5.17.

The ROM-Tree model shows three series system groups (*Input Systems*, *AV Systems*, and *Power Systems*) and one parallel system group (*Memory Systems*). These system groups are represented using stacked rectangles. The *Power Systems* group and its subsystems (*AC Adapter* and *Power Cable)* are represented by dashed boxes, indicating they are modularized and periodically replaceable systems, which our ROM

[19] The structure used here for Notebook PC 390E is based on an actual product (the IBM ThinkPad 390E model) but the numbers are notional and intended for method demonstration purposes only.

method does not include in target reliability allocation because they are interchangeable anytime and have their own target reliability metrics.

The main subsystem of *Notebook PC 390E* is *Main Board Assembly* (Figure 7.6). It is a complex homogenous system that consists of seven simple homogeneous systems (Figure 7.7): *System Board*, *CPU Board*, *Batt Board*, *Interposer Board*, *LVDS Board*, *IR Board*, and *LED Board*.

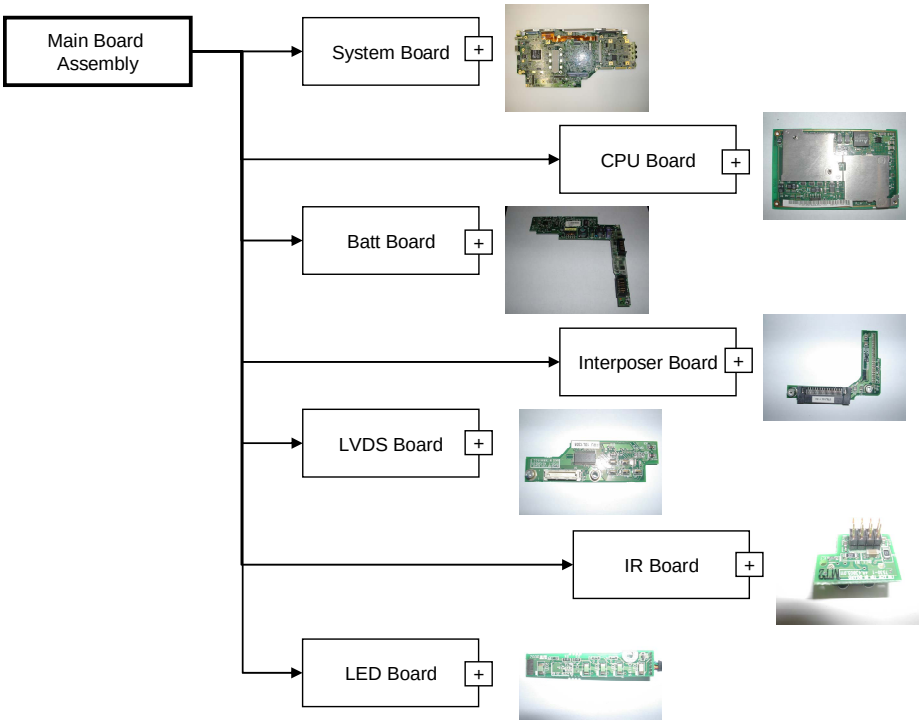


Figure 7.7 A ROM-Tree model for the *Main Board Assembly* subsystems

7.2.3 Target Wearout Failure Reliability Allocation

Similarly to the test case of allocating random failure reliability for video broadcasting system design, the target wearout failure reliability of the notebook PC is allocated following the ROM tree structure in the previous section.

We assume the target wearout failure reliability is allocated from *Notebook PC 390E* to *Main Board Assembly* using historical data-based RTRWs, and the resultant target wearout failure reliability of *Main Board Assembly* is as follows:

$R_w(T_w)_{\text{target, } \textit{Main Board Assembly}} = 0.500$ at $T_w = 45000$ hours

With this target wearout failure reliability, we demonstrate the target wearout failure reliability allocation for series structures. Before target reliability allocation, the RTRWs of subsystems must be setup.

We use the design information-based RTRW ($RTRW^d$). We count the approximate number of components in each board and calculate $ENC_{i,j}$ assuming limited preliminary design information (Table 7.4). For $ECSCC_{i,j}$ calculation, we set guide lines for the PWBA domain from experience: the complexity of BGA chip packages to be 10. For example, since the system board includes three BGA packages, the expected complexity sum is 30 (Table 7.4).

After the calculation of $ENC_{i,j}$ and $ECSCC_{i,j}$, we calculate $TENC_i$ that is the sum of all $ENC_{i,j}$ and $TECSCC_i$ that is the sum of all $ECSCC_{i,j}$. From these, we calculate all $RENC_{i,j}$ and $RECSCC_{i,j}$. With the assumption of w_a and w_b to be equal (i.e., $w_a = w_b = 0.5$), we calculate $RTRW_{i,j}$. The results are shown in Table 7.4.

Table 7.4 RTRWs for the *Main Board Assembly* subsystems

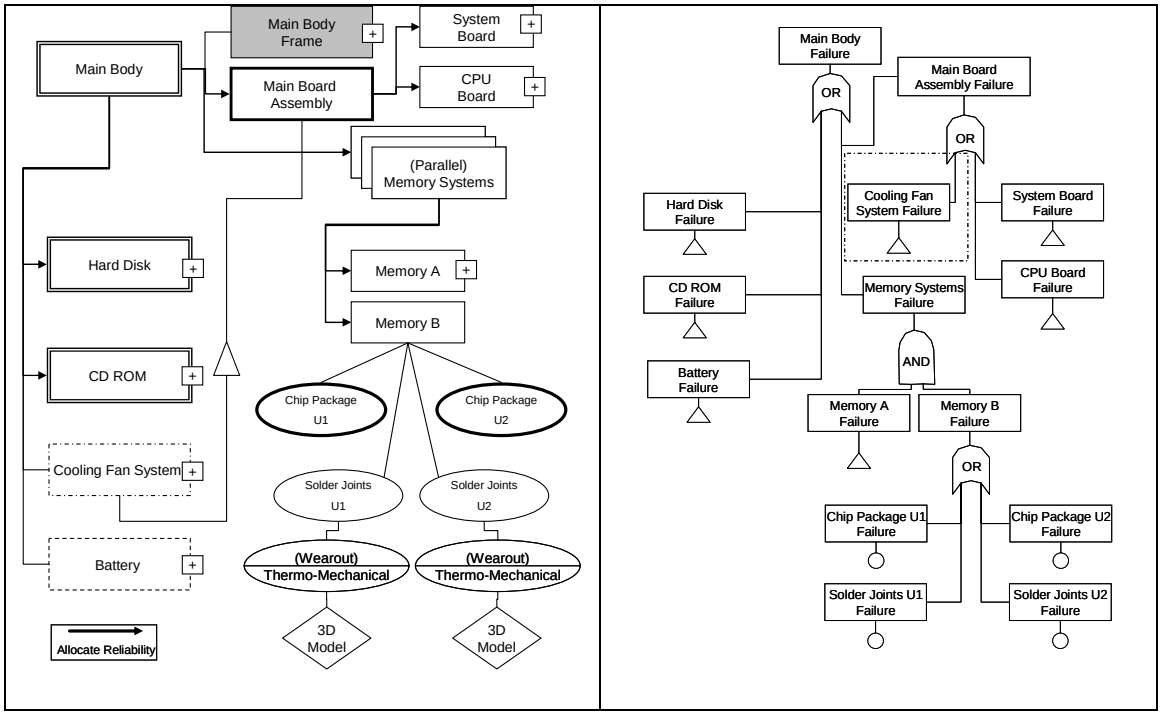
Subsystem Name	RENC	RECSCC	RTRW[d]
1.1.2.1 System Board	600/820 =0.732	30/50 =0.600	0.5×0.732+0.5×0.600 =0.666
1.1.2.2 CPU Board	100/820 =0.122	20/50 =0.400	0.5×0.122+0.5×0.400 =0.261
1.1.2.3 BATT Board	80/820 =0.098	0/50 =0.000	0.5×0.098+0.5×0.000 =0.049
1.1.2.4 Interposer Board	10/820 =0.012	0/50 =0.000	0.5×0.012+0.5×0.000 =0.006
1.1.2.5 LVDS Board	10/820 =0.012	0/50 =0.000	0.5×0.012+0.5×0.000 =0.006
1.1.2.6 IR Board	10/820 =0.012	0/50 =0.000	0.5×0. 012+0.5×0.000 =0.006
1.1.2.7 LED Board	10/820 =0.012	0/50 =0.000	0.5×0. 012+0.5×0.000 =0.006
TENC = 820 (= 600 + 100 + 80 + 10 + 10 + 10 + 10), TECSCC = 50 (= 30 + 20 + 0 + 0 + 0 + 0 + 0).			

Table 7.5 Target reliability allocation results for the *Main Board Assembly* subsystems

Subsystem name (RTRW)		$R_w(T_w)_{target}$
1.1.2 Main Board Assembly		0.500
	1.1.2.1 System Board (0.666)	0.630
	1.1.2.2 CPU Board (0.261)	0.835
	1.1.2.3 BATT Board (0.048)	0.967
	1.1.2.4 Interposer Board (0.006)	0.996
	1.1.2.5 LVDS Board (0.006)	0.996
	1.1.2.6 IR Board (0.006)	0.996
	1.1.2.7 LED Board (0.006)	0.996

Per the RTRWs calculated in Table 7.4, the target wearout failure reliability of the main board assembly is allocated to its subsystems. For example, the target wearout failure reliability of the system board is the target wearout failure reliability of the main board assembly to the power of its RTRW (Equation (5.37)). Accordingly, the allocated target reliability for the system board is $R_w(T_w)_{target\ i.1}$=0.630 at T_w= 45000 hours. The complete results are shown in Table 7.5.

7.2.4 Discussion



(a) ROM-Tree model (b) FTA model

Figure 7.8 Comparing ROM-Tree vs. FTA models for target reliability allocation

In addition to the demonstration of the ROM method for target wearout failure reliability allocation in the previous sections, the effectiveness of the ROM method is also shown by a comparison between a ROM-Tree model and a model created by existing reliability analysis methods. Figure 7.8 shows a ROM-Tree model and a FTA model for the target reliability allocation of *Notebook PC 390E*. The ROM-Tree version shows non-homogeneous systems, a complex homogeneous system, simple homogeneous systems, a modularized system, an auxiliary system, an ignorable system, a parallel system group, commercial components, designed components, failure modes, and prediction models. In addition, the ROM-Tree shows the target reliability allocation activities with arrow lines, reliability composition as lines, and the failure dependency as

a blank triangle. While the ROM-Tree model shows this extensive information in a concise way, the FTA model (Figure 7.8-(b)) shows only parallel and series structures of failure events. Therefore, with the FTA model, we cannot distinguish systems, components, failure modes, and there is no mechanism to allocate target reliability.

Furthermore, *Cooling Fan System* failure is one possible cause of *Main Board Assembly* failure. While this failure relationship is captured in both models, the FTA model cannot represent the associated assembly structure. Thus, the reliability engineer may not realize the fan is a part of the *Main Body*, which may lead to erroneous conclusions for system design changes. Such situations demonstrate how ROM-Tree model richness will likely enable better reliability design decisions.

The reliability allocation for the notebook PC is also demonstrated through the prototype CASDfR tools. Figure 7.9 illustrates reliability objects of the notebook PC in GT-SDfR-Manager, and Figure 7.10 show reliability objects of the notebook PC in P21 format [Kemmerer, 1999].

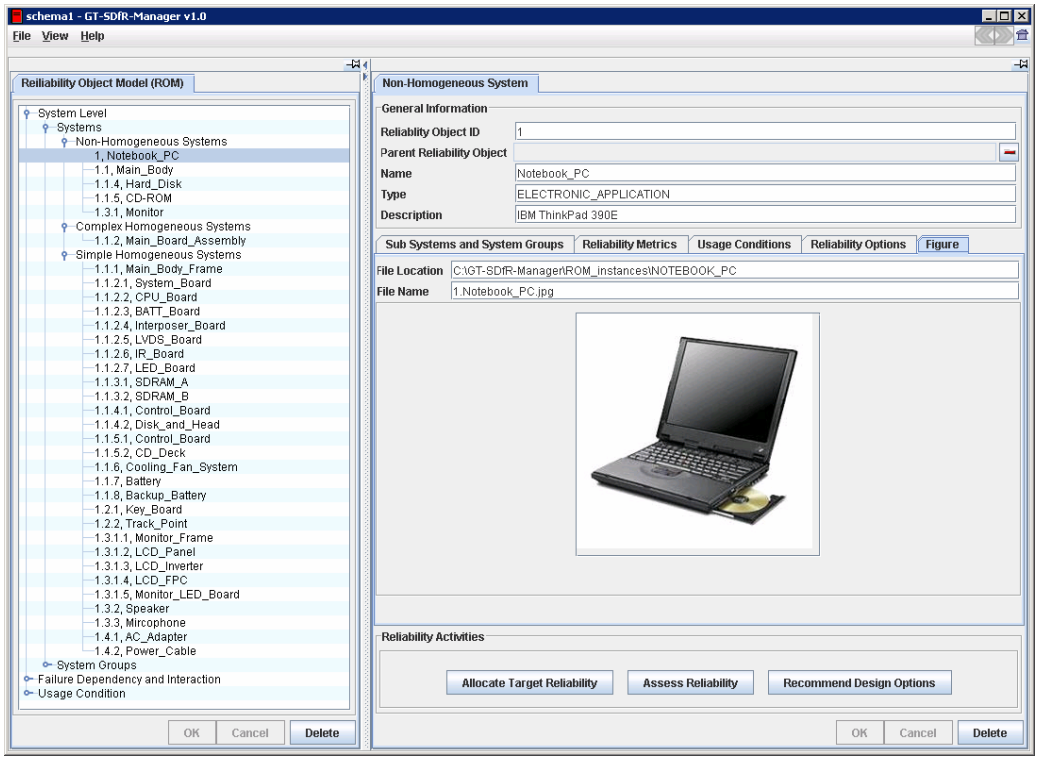


Figure 7.9 Reliability objects of *Notebook PC 390E* design implemented in GT-SDfR-Manager

```
ISO-10303-21;
HEADER;
FILE_DESCRIPTION((''),'');
FILE_NAME('','',(''),(''),'','','');
FILE_SCHEMA(('GT_SDFR_MANAGER'));
ENDSEC;
DATA;
#1=NON_HOMOGENEOUS_SYSTEM($,'1',5773.0,45000.0,0.3404464984464268,.NO.,'Notebook_PC',
    'ELECTRONIC_APPLICATION','IBM ThinkPad 390E',5000.0,45000.0,0.5,$,$,.NO.,.NO.,
    $,#229);
#4=NON_HOMOGENEOUS_SYSTEM(#1,'1.1',5273.0,45000.0,0.3648815224482836,.NO.,'Main_Body',
    'ELECTRONIC_SYSTEM',$,4500.0,45000.0,0.5358867312681466,0.9,0.9,.NO.,.NO.,$,
    #229);

#135=COMPLEX_HOMOGENEOUS_SYSTEM(#4,'1.1.2',4148.0,45000.0,0.4264652218579432,.NO.,
    'Main_Board_Assembly','ELECTRONIC_SYSTEM',$,3375.0,45000.0,0.626332219312064,0.75,0.75,
    .NO.,.NO.,$,#229);
#138=PARALLEL_SYSTEM_GROUP(#4,'1.1.3',225.01447,45000.0,0.969375,.NO.,'Memory_System',
    'ELECTRONIC_SYSTEM',$,225.0,45000.0,0.969289816935065,$,0.05,0.05,1,.ACTIVE.,
    .HOMOGENEOUS.);
#141=NON_HOMOGENEOUS_SYSTEM(#4,'1.1.4',450.0,45000.0,0.9395227492140118,.NO.,'Hard_Disk',
    'ELECTRO-MECHANICAL_SYSTEM',$,450.0,45000.0,0.9395227492140118,0.1,0.1,.NO.,
    .NO.,$,#229);
#144=NON_HOMOGENEOUS_SYSTEM(#4,'1.1.5',450.0,45000.0,0.9395227492140118,.NO.,'CD-ROM',
    'ELECTRO-MECHANICAL_SYSTEM',$,450.0,45000.0,0.9395227492140118,0.1,0.1,.NO.,
    .NO.,$,#229);
#150=SIMPLE_HOMOGENEOUS_SYSTEM(#4,'1.1.6',$,$,$,.NO.,'Cooling_Fan_System',
    'MECHANICAL_SYSTEM',$,$,$,$,$,$,.NO.,.YES.,$,#229);
```

Figure 7.10 Reliability objects of *Notebook PC 390E* design in P21 format

7.3 ROM-BASED RELIABILITY PREDICTION AND ASSESSMENT FOR USB HUB DESIGN

7.3.1 Overview of a USB Hub Test Case

Figure 7.11 A representative USB hub (GE™ USB Hub, UH514)

A representative USB hub — GE™ USB Hub UH514 (Figure 7.11) — is chosen for demonstrating the ROM method for reliability prediction and assessment. Rationales for selection of this test case are as follows:

- The reliability of USB ports has been problematic.
- USB hubs consist of more than 100 commercial components and parallel component groups, so they are complex enough to use the ROM method.
- USB hubs show a clear assembly structure.
- Design information of USB hubs is available from direct measurement.

The target reliability of the USB hub is set up as λ_{target} = 1000 FIT, $R_w(T_w)_{target}$ = 0.5, and T_w = 60000 hours. Besides, two different usage conditions are modeled. One is *Ground Benign* condition [DoD, 1991] (e.g., a temperature-controlled computer room

condition), and the other is *Ground Mobile* condition [DoD, 1991] (e.g., a condition inside automobiles). Depending on usage conditions, temperature range and frequency differ. Under *Ground Benign* condition, USB temperature changes from 20 °C to 55 °C. Under *Ground Mobile* condition, USB temperature changes from 5 °C to 90 °C. The details of two usage conditions are described in Figure 7.12.

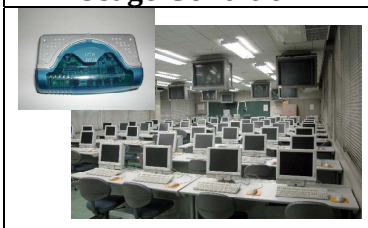
	Usage Condition A	**Usage Condition B**
		
Environment Class	Ground Benign	Ground Mobile
Temperature	20 °C to 55 °C	5 °C to 90 °C
Duty Operation Ratio	25%	25%
Frequency	Three times a day	One time a day

Figure 7.12 Usage conditions of USB hubs

7.3.2 ROM-Tree Model for a Representative USB Hub

Figure 7.13 shows a ROM-Tree model for the USB hub. This model starts from the top level system, *USB Hub 514*, which includes mechanical and electrical subsystems together. Therefore, it is represented with a double-line box. The *Case Assembly* system belongs to the mechanical structure design domain, and the *PWBA S4143786* system belongs to the PWBA design domain.

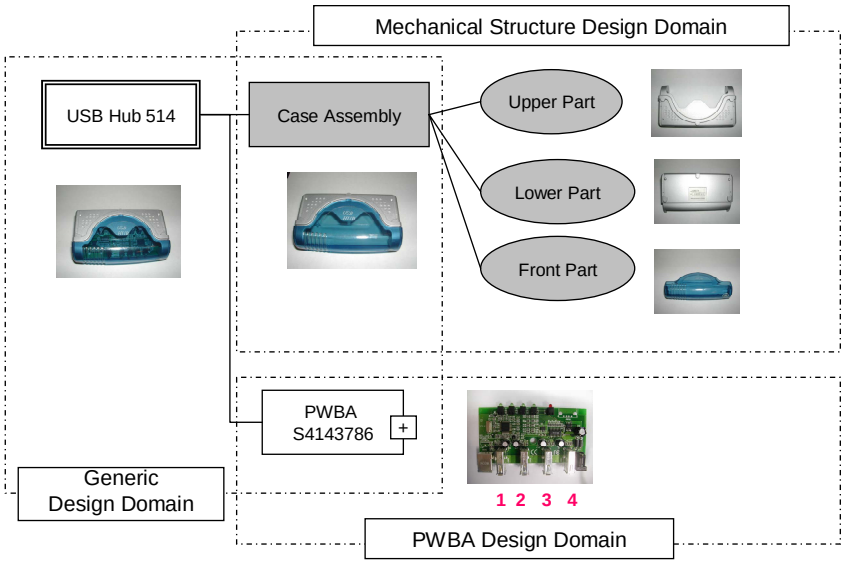


Figure 7.13 A ROM-Tree model for a representative USB hub[20]

A ROM-Tree model for sub components of *PWBA S4143786* is represented in Figure 7.14. The *PWBA S4143786* system operates under *ground benign usage condition*, which is represented by a parallelogram, and includes a parallel port group, a control series group, a PTHs parallel group, and a board. The parallel port group is a 2-out-of-4 parallel component group because at least two ports out of four should work as a hub according to the design specifications. Each port is represented by a series component group, and it includes six sub series component groups.

Group 2.6 in Figure 7.14 is designed for failure monitoring of *2nd Port*. The dot-dashed sacked ovals of *Group 2.6* indicate it is modeled as an auxiliary component group, which is not directly related to the main functionalities of the top level system but important for reliability. Our ROM method does not include auxiliary component groups for reliability assessment. However, the reliability of the auxiliary component group is

[20] The structure used here for USB Hub 514 is based on an actual product (GE UH 514 model). However the reliability numbers we use are not actual values and are intended for method demonstration purposes only.

related to the reliability of associated component groups. For example, the reliability of *Group 2.6* must be greater than the reliability of *2nd Port*, which is represented by the failure dependency blank triangle in Figure 7.14.

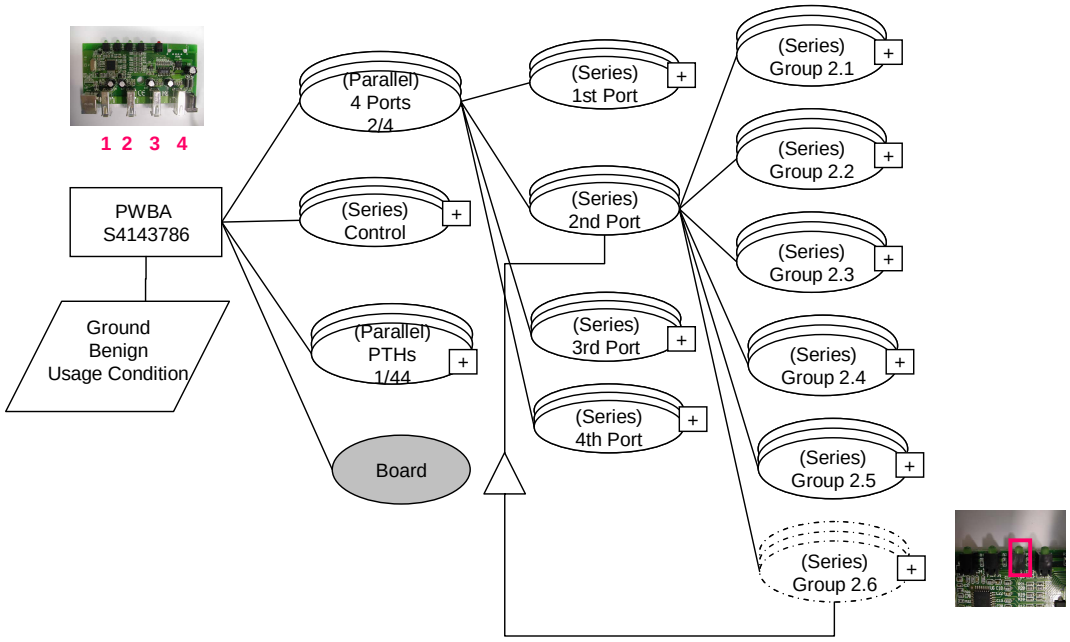


Figure 7.14 A ROM-Tree model for PWBA S4143786

Links between the ROM-Structure of *PWBA S4143786* and the ROM-Library in the PWBA design domain is illustrated in Figure 7.15. In *Group 2.1*, five components (one *jack*, two *capacitors*, and two *inductors*) are serially connected in Figure 7.15. Their reliability is predicted by the MIL-HDBK-217 data [DoD, 1991], which are represented by circles. These components are connected to the board by solder interconnections that may fail by thermo-mechanical fatigue loads. Therefore, their reliability is predicted by solder joint thermo-mechanical fatigue models, which are represented by diamonds.

Design features that are necessary for reliability prediction are represented by large rectangular boxes with small rectangular boxes around them. For example, the design features of solder joints include five parameters: *standoff height*, *fillet height*,

length, *width*, and *solder joint material*. Each parameter has its own value, but the value is not displayed in Figure 7.15.

Some components such as the *jack* and the *solder fillets* are not significant from the reliability point of view, so they are considered ignorable for reliability prediction and assessment, which are represented by gray-filled ovals.

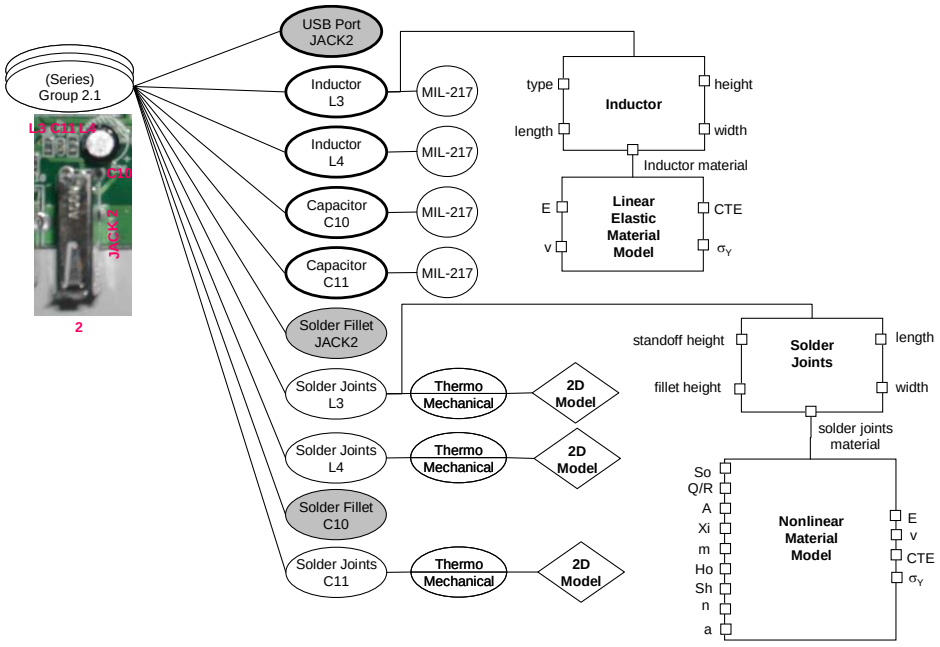


Figure 7.15 A ROM-Tree model for *Group 2.1* of *PWBA S4143786*

All reliability objects in the USB hub electronic board are summarized in Table 7.6 and Table 7.7. The former summarizes the reliability objects from the structure perspective, and the latter summarizes the reliability objects from the component perspective.

Table 7.6 Reliability objects in *USB hub 514* from the structure perspective

	Number of Total Components	Number of Ignorable Components	Number of Countable Components
1.2 PWBA	351	51	300
1.2.1 Parallel Group of 4 Ports	148	12	136
1.2.1.1 1st Port	37	3	34
1.2.1.2 2nd Port	37	3	34
1.2.1.3 3rd Port	37	3	34
1.2.1.4 4th Port	37	3	34
1.2.2 Series Group of Components	159	39	120
1.2.3 Parallel Group of PTHs	44	0	44

Table 7.7 Reliability objects in *USB hub 514* from the component perspective

Items	Number of Items
Commercial Components	101
Resistors	40
Capacitors	32
Inductors	11
Transistors	2
IC Packages	4
Miscellaneous Items (LEDs, Crystals, Jacks, and so on)	12
Designed Components	250
Solder Fillets	35
Solder Joints	76
Plated Through Holes	139
Series Groups	21
Parallel Groups	2

7.3.3 Reliability Prediction

Since various types of components exist in a system, both random and wearout failures should be considered. In this example, MIL-HDBK-217 reliability data are used for predicting random failure reliability of 101 commercial components. Thermo-mechanical models[21] for solder joints and plated through holes (PTHs) are developed for predicting wearout failure reliability of 250 interconnection components.

7.3.3.1 Random Failure Reliability Prediction

Among various statistics-based reliability prediction models [Denson, 1998; Foucher et al., 2002] introduced in Chapter 3, MIL-HDBK-217 [DoD, 1991] reliability data are used in this example because the handbook data are easily accessible. The handbook is intended to provide consistent and uniform methods for estimating the inherent reliability of electronic equipment and systems [DoD, 1991].

Table 7.8 shows the results of predicting the random failure rates of the commercial components from MIL-HDBK-217 data.

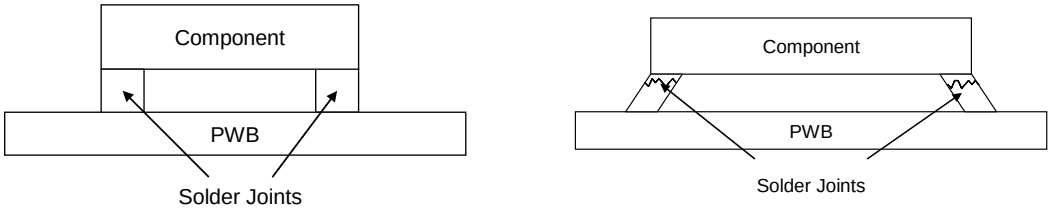
[21] Reliability prediction models are described in Appendix C.

Table 7.8 Random failure reliability prediction results of the *PWBA S4143786* components

The First Category	The Second Category	Number of Components	Condition A $\lambda_{predicted}$ (FIT)	Condition B $\lambda_{predicted}$ (FIT)
Resistor	Film Chip	38	3.7	70
Resistor	Wire wound	2	6.5	160
Capacitor	Ceramic Chip	28	3.5	130
Capacitor	Aluminum Electrolytic	4	1.3	47
Inductor	Ceramic Chip	11	0.03	0.5
Transistor	GaAs FET	2	52	260
IC - MOS	Linear 4Pins	1	9.5	39
IC - MOS	Switch 8Pins	1	5.7	27
IC - MOS	Switch 16Pins	1	5.7	27
IC - MOS	Logic 32Pins	1	19	80
Quartz - Crystals		1	32	320

7.3.3.2 Wearout Failure Reliability Prediction

For predicting the wearout failure reliability of 76 solder joints and 139 PTHs, a thermo-mechanical solder joint fatigue model and a PTH fatigue models are developed.

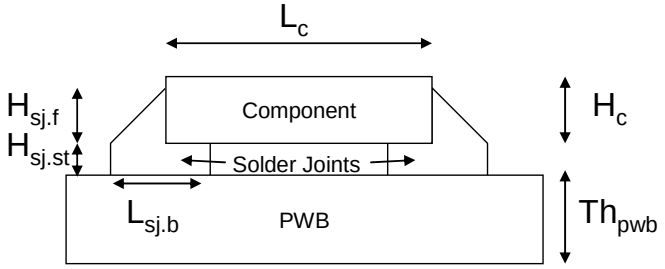


(a) Undeformed State (b) Deformed State

Figure 7.16 Thermo-mechanical fatigue failure of solder joints

Solder joints are interconnection components that connect commercial components and the printed wiring board (PWB). The failure of solder joints is caused by the different thermal expansion between the component and the PWB, illustrated in Figure 7.16. The prediction of thermo-mechanical solder joint fatigue failure requires a finite element analysis (FEA) model to predict cyclic strains and a fatigue model to predict the number of cycles to 50% failure.

The first step of developing the FEA model for solder joints is to identify solder joint design features. The solder joint design features are *solder joint standoff height*, *fillet height*, *base length*, and *solder joint material*. The features are illustrated in Figure 7.17.



where

$H_{sj.st}$: Solder joint standoff height,

$H_{sj.f}$: Solder joint fillet height,

$L_{sj.b}$: Solder joint base length,

H_c : Component height,

L_c : Component length,

Th_{pwb}: PWB thickness.

Figure 7.17 Solder joint design features

Figure 7.18 illustrates a half symmetric plane strain FEA model for solder joints. This model uses the PLANE82 element type of ANSYS and the isotropic material model

for component modeling; the PLANE82 element type of ANSYS and the orthotropic material model for PWB modeling; and the VISCO108 element type of ANSYS and Anand's viscoplastic material model for solder joint modeling.

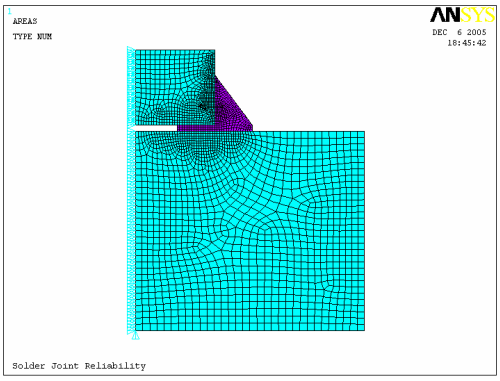
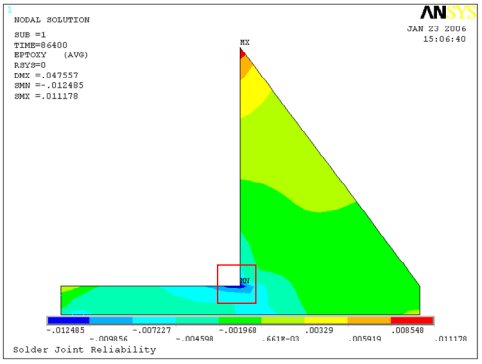
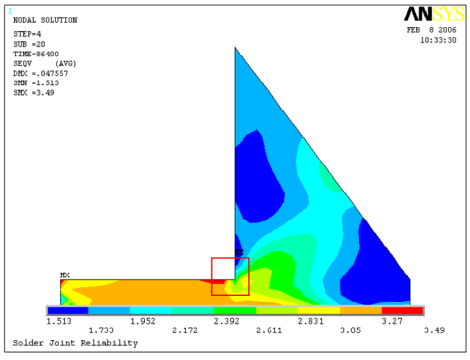


Figure 7.18 A thermo-mechanical fatigue model for solder joints

The test results obtained using this FEA model under a cyclic temperature change between 20°C and 100°C are illustrated in Figure 7.19. The boxes at the corner of the solder joint indicate the area where a crack originated.



(a) Total shear strain



(b) Von Mises stress

Figure 7.19 Simulation results of thermo-mechanical solder joint fatigue

The averaged total shear strain change in the box is a damage metric for predicting the number of cycles to 50% failure (N_{50}). For this N_{50}, Engelmaier's modified Coffin-Manson equation for solder [Engelmaier, 1983] is used, shown in Table 7.9. The simulation results are also shown in Table 7.10. Finally, for the validation of this FEA model, the simulation results are compared with those of Lau, et al. [Lau et al., 1986], shown in Table 7.11. Since a good correlation is observed from both the results, we think that this FEA model is valid and applicable for system reliability prediction.

Table 7.9 The Engelmaier's modified Coffin-Manson equation for solder

$N_{50} = 0.5[\frac{\Delta\gamma_t}{2\varepsilon_f'}]^{1/c_{fde}}$	N_{50} : Number of cycles to 50% failure
	$\Delta\gamma_t$: Averaged total strain change
	ε_f' : Fatigue ductility coefficient
	c_{fde} : Fatigue ductility exponent

Table 7.10 Test results of the thermo-mechanical fatigue model for solder joints

Temperature	**20 °C- 90°C**	**20 °C- 100°C**	**20 °C- 120°C**
Strain	0.0065	0.0079	0.0110
N_{50}	13177	8516	4159

Table 7.11 Validation of the thermo-mechanical fatigue model for solder joints

	Results of Lau, et al.	**Results of the FEA Model**
Strain Change (-55 °C to 125 °C)	0.0143	0.0162

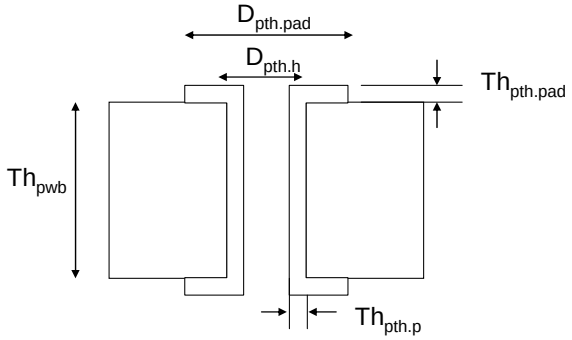
Plated through holes (PTHs) are interconnection components that connect copper traces in different layers. A PTH failure is caused by the different thermal expansion between the PTH and the PWB is illustrated in Figure 7.20.

(a) Undeformed State (b) Deformed State

Figure 7.20 Thermo-mechanical fatigue failure of PTHs

The prediction of thermo-mechanical PTH failure requires a finite element analysis (FEA) model to calculate cyclic strains and a fatigue model to calculate the number of cycles to 50% failure.

The first step of developing the FEA model for thermo-mechanical PTH fatigue failure is to identify PTH design features. The PTH design features are *pad diameter*, *hole diameter*, *pad thickness*, *plating thickness*, and *PTH material*. These features are illustrated in Figure 7.21.



where

$D_{pth.pad}$: Pad diameter,

$D_{pth.h}$: Hole diameter,

$Th_{pth.pad}$: Pad thickness,

$Th_{pth.p}$: Plating thickness,

Th_{pwb}: PWB thickness.

Figure 7.21 PTH design features

Figure 7.22 illustrates an axisymmetric FEA model for PTHs. This model uses the PLANE42 element type of ANSYS and the multilinear kinematic hardening material model for PTH modeling; and the PLANE82 element type of ANSYS and the orthotropic material model for PWB modeling.

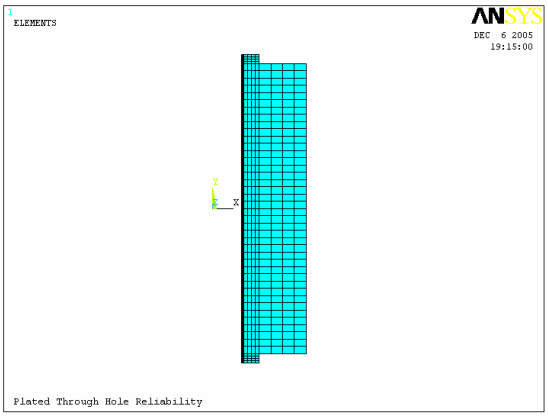
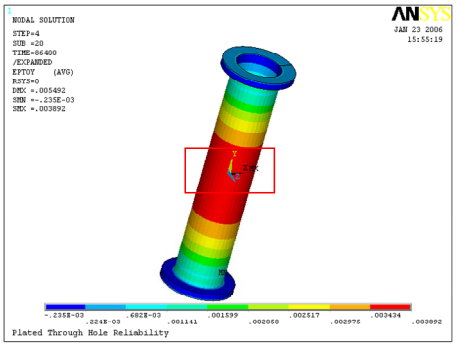
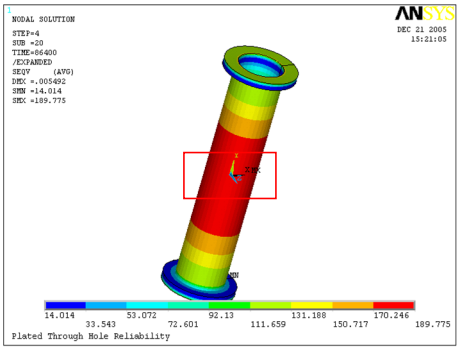


Figure 7.22 A thermo-mechanical fatigue model for PTHs

The simulation results obtained using this FEA model under a cyclic temperature change between 20°C and 100°C are illustrated in Figure 7.23. The boxes at the center indicate the area where a crack originated.



(a) Total strain



(b) Von Mises stress

Figure 7.23 Simulation results of thermo-mechanical PTH fatigue

The average total strain change in the box is a damage metric for predicting the number of cycles to 50% failure (N_{50}). For the calculation of this N_{50}, Engelmaier's modified Coffin-Manson equation for copper [Engelmaier, 1987] is used, shown in Table 7.12. The simulation results are also shown in Table 7.13. The strain at the 20 °C - 90°C temperature change is relatively small because deformation occurs in the elastic region and below the glass transition temperature (110°C), which causes significant thermal expansion of the PWB. For the validation of this FEA model, the simulation results are compared with those of the IPC-TP-510 analytical model [IPC-TP-510, 1984], as shown in Table 7.14. Since a good correlation is observed from both the results, we think that this FEA model is valid and applicable for system reliability prediction.

Table 7.12 The Engelmaier's modified Coffin-Manson equation for copper

$\Delta\varepsilon_t = N_{50}^{-0.6}(\varepsilon_f')^{0.75} + \frac{0.9\sigma_u}{E}[\frac{e^{\varepsilon_f'}}{0.36}]^{0.1785\log(\frac{10^5}{N_{50}})}$	N_{50} : Number of cycles to 50% failure
	$\Delta\varepsilon_t$: Averaged total strain change
	ε_f' : Fatigue ductility coefficient
	σ_u : Ultimate tensile strength
	E : Young's modulus

Table 7.13 Test results of the thermo-mechanical fatigue model for PTHs

Temperature	**20 °C- 90°C**	**20 °C- 100°C**	**20 °C- 120°C**
Strain	0.0012	0.0039	0.0120
N_{50}	3720900	8514	122

Table 7.14 Validation of the thermo-mechanical fatigue model for PTHs

	Result of the IPC Model	**Result of the FEA Model**
Strain Change (20 °C to 100 °C)	0.0024	0.0028

In this USB hub test case, five types of design features are identified, illustrated in Figure 7.24. These features are repeated in the USB board system. Detail parameters of these features are shown in Table 7.15.

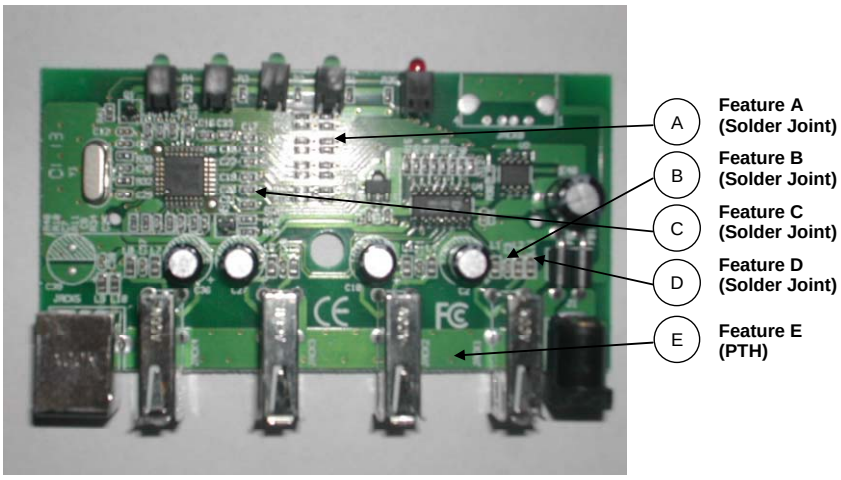


Figure 7.24 *PWBA S4143786* design features

Table 7.15 *PWBA S4143786* design features

		Design Features (Unit: mm)				
		A	**B**	**C**	**D**	**E**
Solder Joint						
	$H_{sj.st}$	0.1270	0.1270	0.2000	0.2000	
	$H_{sj.f}$	0.4000	0.4000	0.7000	0.5000	
	$L_{sj.b}$	0.3000	0.3000	0.3000	0.3000	
	H_c	0.5000	0.7000	0.9000	0.7000	
	L_c	2.0000	2.0000	2.000	2.0000	
	Solder Material	Sn(96.5)-Ag(3.5)	Sn(96.5)-Ag(3.5)	Sn(96.5)-Ag(3.5)	Sn(96.5)-Ag(3.5)	
	Component Material	Alumina (Resistor)	Alumina (Capacitor)	Alumina (Capacitor)	Alumina (Inductor)	
PTH						
	$D_{PTH.pad}$					0.5080
	$D_{PTH.h}$					0.3429
	$Th_{PTH.pad}$					0.0483
	$Th_{PTH.p}$					0.0127
	PTH Material					Copper
PWB						
	Th_b	2.0000	2.0000	2.0000	2.0000	2.0000
	PWB Material	FR4	FR4	FR4	FR4	FR4

Once the parameters of each design feature are applied to the solder joint fatigue model and the PTH fatigue model in the previous section, the wearout failure reliability of each interconnection was predicted. Table 7.16 shows the results. The shape parameters of the Weibull curve (β *of* $R(t)=e^{-(\frac{t}{\alpha})^{\beta}}$) are assumed from experimental data. For the thermo-mechanical fatigue failure of solder joints, the shape parameter of the Weibull curve is 2 [Blattau and Hillman, 2005], and for the thermo-mechanical fatigue failure of PTHs, the shape parameter of the Weibull curve is 5 [Mercado-Corujo, 2001; Suzuki, 2005]. With these shape parameters and $N_{50,hours}$, the characteristic lifetime parameters of the Weibull curve (α *of* $R(t)=e^{-(\frac{t}{\alpha})^{\beta}}$) are calculated from Equation (7.1).

$$\alpha = \frac{N_{50,hours}}{\{-ln(0.5)\}^{\frac{1}{\beta}}} \tag{7.1}$$

Table 7.16 Wearout failure reliability prediction results of the *PWBA S4143786* design features

Design Features	Condition A (Ground Benign)			Condition B (Ground Mobile)		
	Strain	N_{50} [a]	$N_{50,Hours}$ [b]	Strain	N_{50} [a]	$N_{50,hours}$ [b]
A	1.53e-3	5.5197e5	4.4158e6	4.98e-3	2.5072e4	6.0173e5
B	1.53e-3	5.5197e5	4.4158e6	4.98e-3	2.5072e4	6.0173e5
C	1.33e-3	7.6369e5	6.1095e6	3.78e-3	4.6252e4	1.1100e6
D	1.19e-3	9.8368e5	7.8694e6	3.41e-3	5.8067e4	1.3936e6
E	0.70e-3	4.6051e8	3.6841e9	2.03e-3	6.0229e4	1.4455e6
a N_{50} : Number of cycles to 50% failure, b $N_{50,hours}$: N_{50} × Frequency.						

The Weibull functions of each feature are illustrated in Figure 7.25 and Figure 7.26. The comparison of the reliabilities of different features is important for reliability improvement. According to Figure 7.25 and Figure 7.26, features A and B are less

reliable than other features (features C, D, and E), and the feature E (PTH) is more sensitive to usage conditions than other features (features A,B,C, and D). If design changes in the USB hub are necessary because of unsatisfied wearout failure reliability, a reasonable approach would be to change the parameters of features A and B.

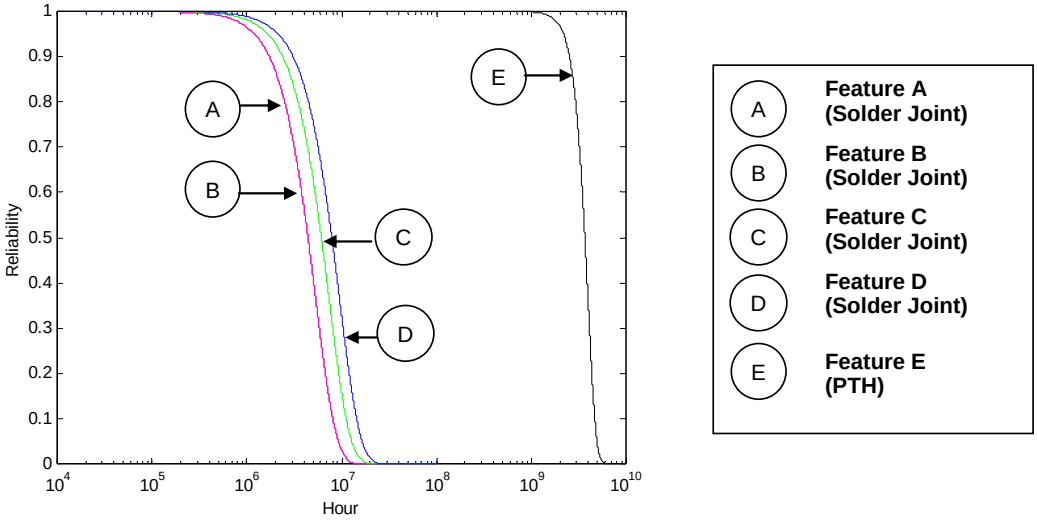


Figure 7.25 Wearout failure reliability prediction results of *PWBA S4143786* design features under condition A

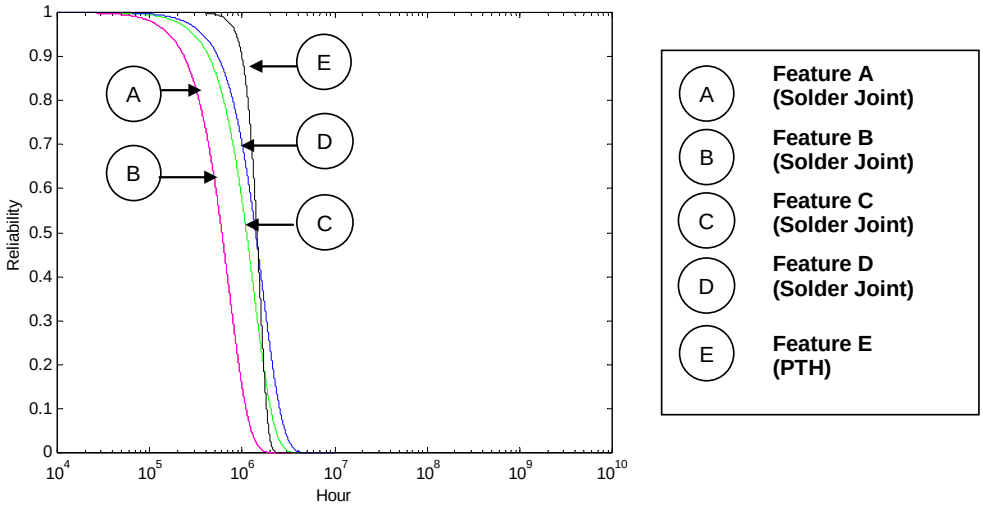


Figure 7.26 Wearout failure reliability prediction results of *PWBA S4143786* design features under condition B

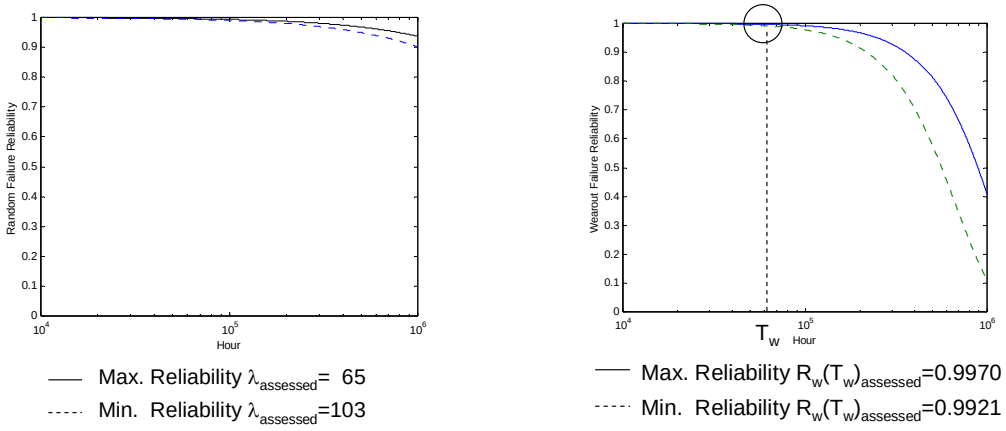
7.3.4 Reliability Assessment

The reliability data predicted in the previous sections are used for assessing the reliability of the USB hub considering the logical structures defined in Section 7.3.2. Since the UBS board system consists of parallel ports, the failure of the USB board system may be interpreted differently depending of the failures of the parallel ports. For example, if the complete functionality of the USB board system is required, the success of all four ports should be. However, if the minimum utility of the USB board system is required, then the success of at lease two ports will be fine. The former that requires the success of all ports is called the minimum reliability of the system, and the latter that requires the success of at least two ports is called maximum reliability of the system.

Both maximum and minimum system reliabilities are assessed from the predicted reliability data under the *Ground Benign* condition. They are shown in Figure 7.27 under a 25% duty operation condition. The maximum system reliability of the USB hub is $\lambda_{assessed}$ = 65.21 FIT and $R_w(T_w)_{assessed}$ = 0.9970 at T_w= 60,000 hours, and the minimum system reliability of the USB hub is $\lambda_{assessed}$ =103.10 FIT and $R_w(T_w)_{assessed}$ =0.9921 at T_w= 60,000 hours.

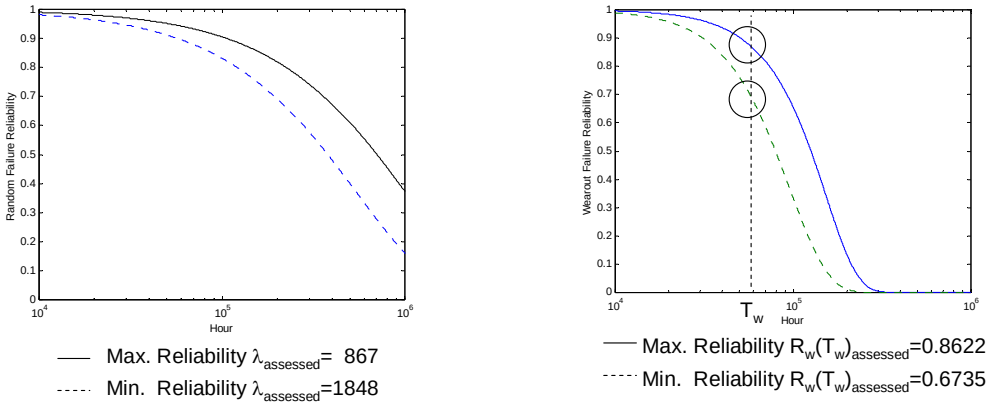
Similar to the assessment under the *Ground Benign* condition, the system reliabilities of the *PWBA S4143786* are assessed under condition B, the *Ground Mobile* condition. The results are shown in Figure 7.28 under a 25% duty operation condition. The maximum system reliability of the USB hub is $\lambda_{assessed}$ =1000 FIT and $R_w(T_w)_{assessed}$ =0.8622 at T_w= 60000 hours, and the minimum system reliability of the USB hub is $\lambda_{assessed}$ =1848 FIT and $R_w(T_w)_{assessed}$ =0.6735 at T_w= 60,000 hours. These results are quite

different from those of the *Ground Benign* condition. This implies that modeling usage conditions is important for correct reliability assessment.



(a) Random failure reliability (b) Wearout failure reliability

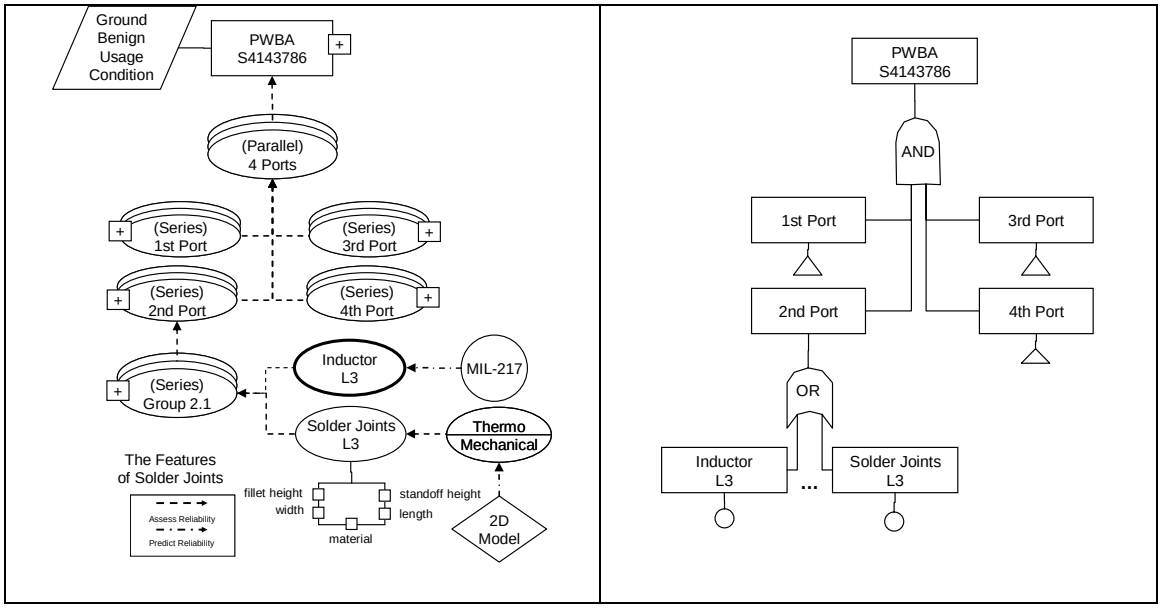
Figure 7.27 *USB Hub 514* reliability assessment results under the condition A



(a)Random failure reliability (b) Wearout failure reliability

Figure 7.28 *USB Hub 514* reliability assessment results under the condition B

7.3.5 Discussion



(a) ROM-Tree model (b) FTA model

Figure 7.29 Comparing ROM-Tree vs. FTA models for reliability prediction and assessment

The previous sections demonstrate the ROM method for reliability prediction and assessment. This demonstration is abbreviated in a ROM-Tree model in Figure 7.29-(a), and it is compared with an equivalent FTA model in Figure 7.29-(b). Figure 7.29-(a) shows a simple homogeneous system, a parallel component group, series component groups, a commercial component, a designed component, a failure mode, a usage condition, prediction models, and design features. In addition, it shows reliability prediction and assessment activities with dashed lines and dashed-dotted lines. Whereas the ROM-Tree model shows the concise information for reliability prediction and assessment, the FTA model shows only parallel and series structure of failure events. Therefore, the use of FTA models for reliability prediction and assessment is limited.

In addition to its qualitative effectiveness shown in Figure 7.29, the ROM method is also quantitatively effective. For example, Table 7.17 summarizes the maximum and minimum reliabilities of *USB hub 514* under two different usage conditions. These results are compared with those of existing Part Count Method. The Part Count Method [Crowe and Feinberg, 2001] sums up random failure rates of all components and does not consider logical structures, ignorable components, failure interactions, and wearout failure reliability. Therefore, the reliability assessment results are limited for design decisions [Jensen, 1995]. However, the ROM method provides richer semantics and complete reliability assessment results for design decisions.

Table 7.17 Summary of *USB hub 514* reliability assessment results

	Condition A (Ground Benign)		**Condition B (Ground Mobile)**	
	$\lambda_{assessed}$ (FIT)	**$R_w(T_w)$ assessed (T_w = 60000 Hrs)**	**$\lambda_{assessed}$ (FIT)**	**$R_w(T_w)$ assessed (T_w = 60000 Hrs)**
Target Reliability	1000	0.5000	1000	0.5000
ROM Method: Max. Reliability Assessment	65	0.9970	867	0.8622
ROM Method: Min. Reliability Assessment	103	0.9921	1848	0.6735
Part Count Method: Reliability Assessment	108	NA	1925	NA

The reliability prediction and assessment for *USB hub 514* are also demonstrated through prototype CASDfR tools. Figure 7.30 illustrates CASDfR-Framework and prototype CASDfR tools for *USB HUB 514* design.

The target reliability of *USB hub 514* is allocated to *PWBA S4143786* through GT-SDfR-Manager (the left side of Figure 7.30). Then, the component reliabilities of *PWBA S4143786* are predicted through MIL-HDBK-217 data and ANSYS^{TM} script

models (the bottom of Figure 7.30). For reliability prediction, GT-SDfR-PWBA creates SQL query statements and simulation script files out of usage condition and design feature information captured in ROM-Tree models. After the simulations are finished, the results are uploaded to GT-SDfR-PWBA from simulation domain tools.

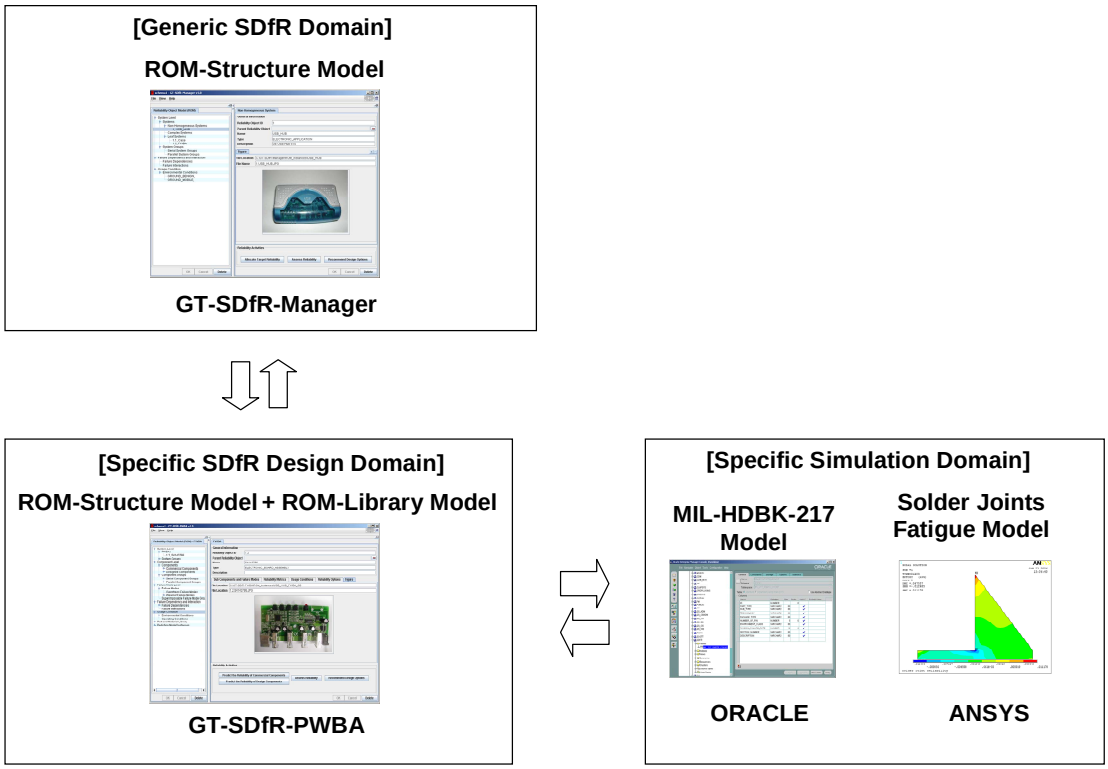


Figure 7.30 CASDfR-Framework and prototype tools for *USB HUB 514* design

7.4 ROM-BASED DESIGN RECOMMENDATIONS FOR ENGINE CONTROL UNIT (ECU) DESIGN

7.4.1 Overview of a Mockup ECU Test Case

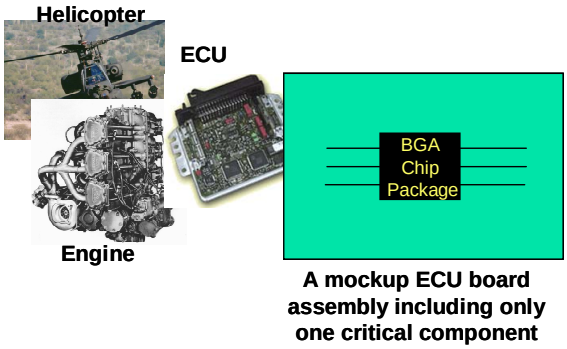


Figure 7.31 A mockup ECU

A mockup helicopter ECU (Figure 7.31) is chosen for demonstrating the ROM method for design recommendation. Rationales for selecting this test case are summarized as follows:

- The reliability of ECUs has been problematic [NTSB, 2006].
- ECUs work under harsh environments, and they experience multiple failure modes.
- The design information of ECUs is available from literature.
- The mockup ECU has only one critical component, so the assembly structure is simple.

The target reliability of the ECU is set up as λ_{target}=250 FIT, $R_w(T_w)_{target}$=0.995, and T_w= 90,000 hours. Besides, the usage condition of the ECU is classified as *Airbone Rotary Winged* condition [DoD, 1991]. The ECU experiences a temperature change from -15 °C to 70 °C and 0.5G level random vibration [Chenault, 2006; Thakkar, 2005]. It works once in two days with 25% duty operation. The usage conditions of the ECU are summarized in Table 7.18.

Table 7.18 Usage conditions of the ECU

Environment Class	Airbone Rotary Winged
Temperature	-15 °C to 70 °C
Duty Operation Ratio	25%
Frequency	Once in two days
Random Vibration Level	0.5G

7.4.2 ROM-Tree Model for a Mockup ECU Board Assembly

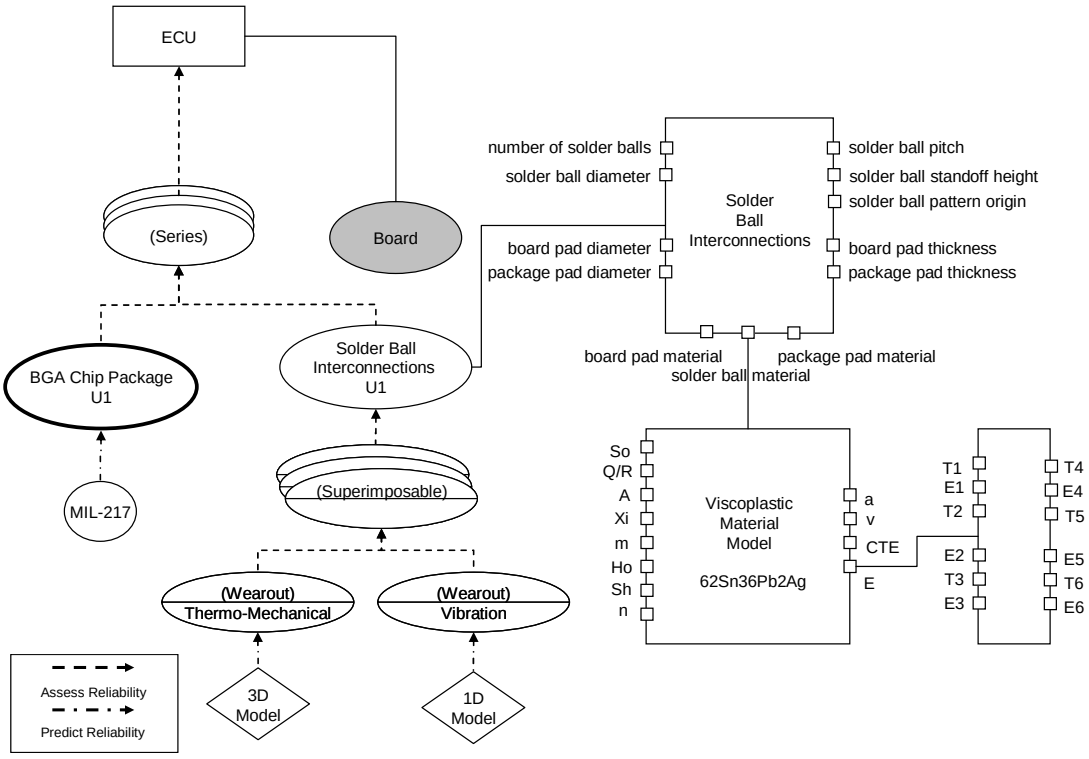


Figure 7.32 A ROM-Tree model for the ECU board assembly

Figure 7.32 shows a ROM-Tree model for the ECU including one critical component, a Ball Grid Array (BGA) chip package. Since helicopters operate in harsh environments, two superimposable failure modes in this ROM-Tree model are modeled: a thermo-mechanical failure mode and a vibration failure mode. In addition, the ROM-Tree model shows design features such as solder ball interconnections and their nonlinear material model.

7.4.3 Reliability Prediction and Assessment

For reliability prediction in this example, the MIL-HDBK-217 data are used for random failures, and while FEA models[22] and analytical models are used for wearout failures.

7.4.3.1 Random failure reliability Prediction

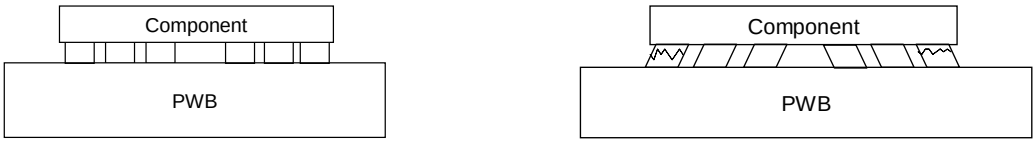
According to the MIL-HDBK-217 data, the random failure rate of the BGA chip package under an *Airbone Rotary Winged* environment is λ = 820 FIT. Since the duty operating ratio is 25%, the actual random failure rate of the BGA chip package is $\lambda_{predicted}$ = 205 FIT. This result satisfies the target reliability, λ_{target} =250 FIT. Therefore, design changes are not recommended for random failure reliability.

7.4.3.2 Wearout failure reliability Prediction

For the wearout failure reliability prediction of the ECU, two failure modes are considered: the thermo-mechanical fatigue failure mode and the vibration-induced fatigue failure mode of solder ball interconnections. The thermo-mechanical fatigue failure of solder ball interconnections is caused by the different thermal expansion between the component and the PWB, illustrated in Figure 7.33. The simulation of thermo-mechanical fatigue failure for solder ball interconnections requires a finite element analysis (FEA) model to predict solder ball cyclic strains and a fatigue model to predict

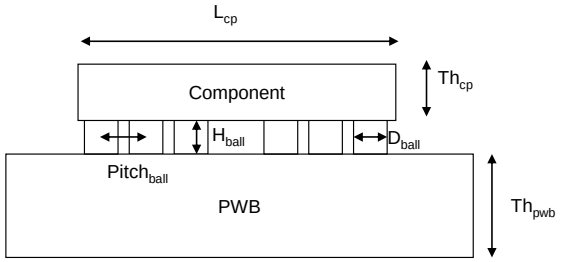
[22] Reliability prediction models are described in Appendix C.

the number of cycles to 50% failure. The first step of the FEA model development is to identify design features of solder ball interconnections. The design features of solder ball interconnections are *solder ball standoff height*, *solder ball diameter*, *solder ball pitch*, *the number of solder balls*, and *solder ball material*. These features are illustrated in Figure 7.34.



(a) Undeformed State (b) Deformed State

Figure 7.33 Thermo-mechanical fatigue failure of solder ball interconnections



where

H_{ball} : Solder ball height,

D_{ball} : Solder ball diameter,

$Pitch_{ball}$: Solder ball pitch,

L_{cp} : BGA chip package length,

Th_{cp}: BGA chip package thickness,

Th_{pwb}: PWB thickness.

Figure 7.34 Solder ball interconnection design features for thermo-mechanical fatigue analysis

Figure 7.35 illustrates the developed FEA model, a half symmetric generalized plane deformation model. This model uses the PLANE45 element type of ANSYS and

the isotropic material model for BGA chip package modeling; the PLANE45 element type of ANSYS and the orthotropic material model for PWB modeling; and the VISCO107 element type of ANSYS and the Anand's viscoplastic material model for solder ball interconnections modeling.

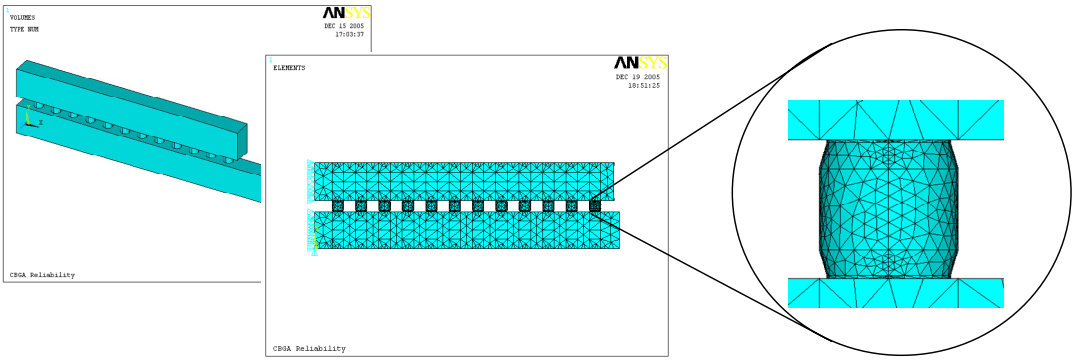
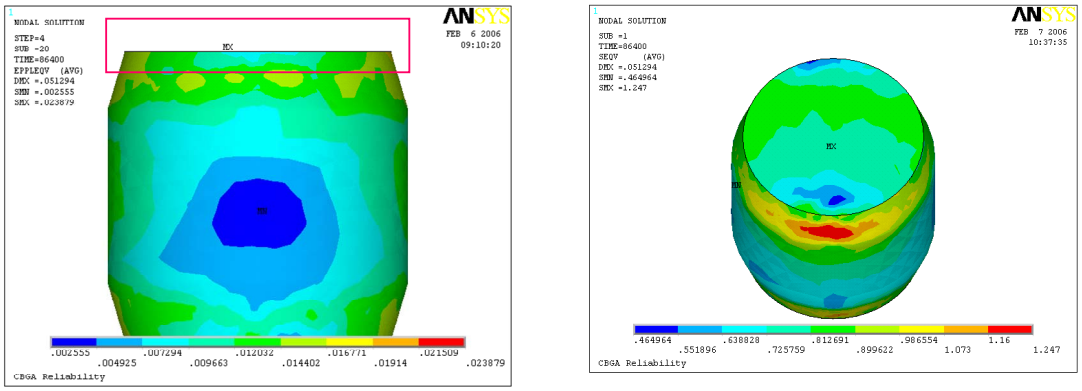


Figure 7.35 A thermo-mechanical fatigue model for solder ball interconnections

The test results obtained using this FEA model at a cyclic temperature change between 20°C and 100°C are illustrated in Figure 7.36. The boxes at the top of the solder ball indicate the area where a crack originated.



(a) Total shear strain (b) Von Mises stress

Figure 7.36 Simulation results of thermo-mechanical solder ball interconnections fatigue

The averaged inelastic strain change in the box is a damage metric for predicting the number of cycles to 50% failure (N_{50}). For the calculation of the N_{50}, a Coffin-

Manson equation for solder ball interconnections is used (Table 7.19) [Perkins and Sitaraman, 2003].

The simulation results are shown in Table 7.20 and compared with the results of a one-dimensional analytical model, shown in Table 7.21. Since the strain results are in good agreement with the analytical model, we think that this FEA simulation model is valid and applicable for system reliability prediction.

Table 7.19 A Coffin-Manson equation for solder ball interconnections

$N_{50} = C_f(\Delta\varepsilon_{in})^{(1/n)}$	N_{50} : Number of cycles to 50% failure
	$\Delta\varepsilon_{in}$: Averaged inelastic strain change
	n : Fatigue exponent coefficient
	C_f : Fatigue coefficient

Table 7.20 Test results of the thermo-mechanical fatigue model for solder ball interconnections

Temperature	**20 °C- 90°C**	**20 °C- 100°C**	**20 °C- 120°C**
Strain	0.0140	0.0160	0.0217
N_{50}	3279	2824	2241

Table 7.21 Validation of the thermo-mechanical fatigue model for solder ball interconnections

	Results of the Analytical Model	**Results of the FEA Model**
Strain Change (20 °C to 120 °C)	0.0237	0.0217

Vibration-induced failures of solder ball interconnections are caused by the tensile stress of solder ball interconnections by PWB bending at a natural frequency (Figure 7.37) [Perkins and Sitaraman, 2004; Steinberg, 2001]. The procedure of predicting vibration-induced reliability of solder ball interconnections is quite different from those of predicting thermo-mechanical reliability. Figure 7.38 summarizes the procedure.

(a) Undeformed State (b) Deformed State

Figure 7.37 Vibration-induced fatigue failure of solder ball interconnections

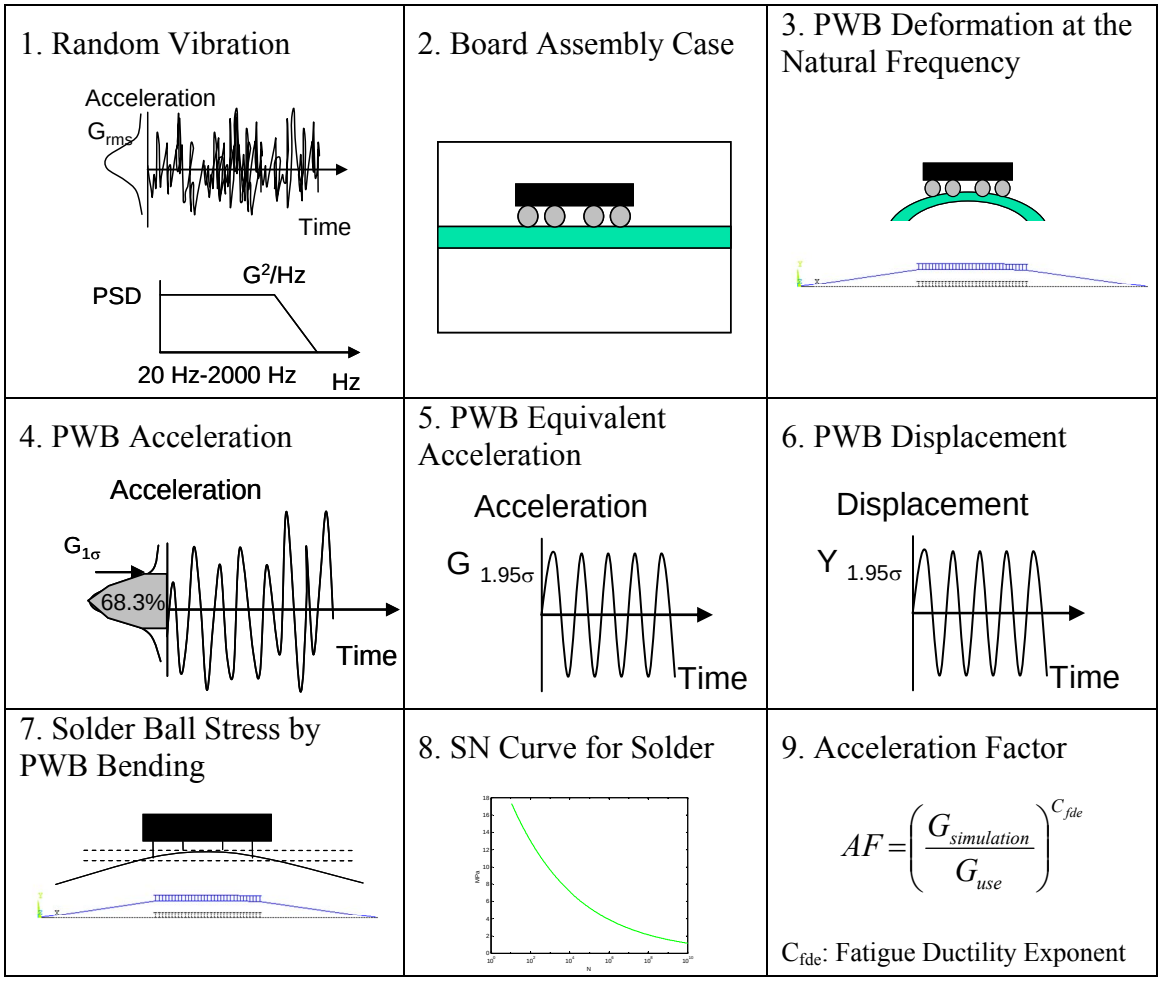


Figure 7.38 Procedures for predicting vibration-induced reliability of solder ball interconnections

The first step of the procedure is to model external vibration sources. Generally speaking, helicopters are excited by random vibration[23], and the acceleration of helicopters is modeled 4 Grms from 20Hz to 2,000 Hz according to MIL-STD-810E [DoD, 1989], illustrated in Figure 7.39.

TABLE 513.4-I Suggested G levels for Procedure I - Structural test

Vehicle Category	Forward Acceleration A in g's 1/	Test Level: Direction of Vehicle Acceleration See Figure 513.4-1					
		Fore	Aft	Up	Down	Lateral	
						Left	Right
Aircraft 2/, 3/	2.0	1.5A	4.5A	6.75A	2.25A	3.0A	3.0A
Helicopters	4/	4.0	4.0	10.5	4.5	6.0	6.0
Manned Aerospace Vehicles	6.0 to 12.0 5/	1.5A	0.5A	2.25A	0.75A	1.0A	1.0A
Aircraft Stores: Wing/Sponson Mounted	2.0	7.5A	7.5A	9.0A	4.9A	5.6A	5.6A
Aircraft Stores: Fuselage	2.0	5.25A	6.0A	6.75A	4.1A	2.25A	2.25A
Ground-Launched Missiles	6/, 8/	1.2A	0.5A	1.2A'	1.2A' 7/	1.2A' 7/	7/

Figure 7.39 Random vibration test specification of helicopters (MIL-STD-810E)

The ECU vibrates due to the helicopter acceleration. This vibration is illustrated at the second and the third steps in Figure 7.38. Since a single-degree-of-freedom spring-mass system excited by random vibration can respond only by vibrating at its natural frequency [Steinberg, 2001], the ECU vibrates at its natural frequency. However, the amplitude of the vibration is still random, which is illustrated in the fourth step in Figure 7.38. The response of the ECU — the vibration at its natural frequency with random amplitudes — can be simplified using the three-band technique for random vibration [Steinberg, 2001]. The equivalent response is sinusoidal vibration at its natural

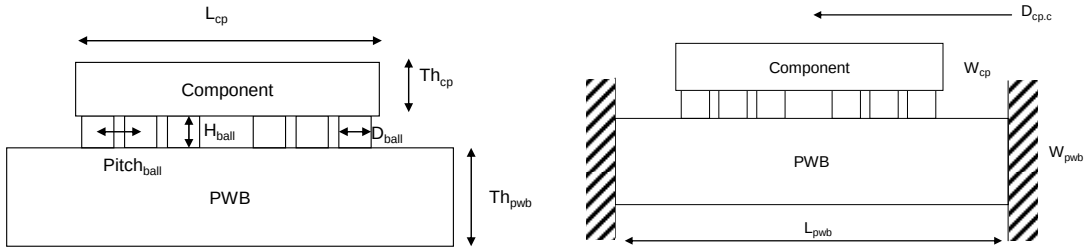
[23] Random vibration consists of normal distributed amplitudes and continuous frequency ranges.

frequency with the uniform amplitude. This response is 1.95 times standard deviation of the normal distributed amplitudes of the random vibration, illustrated in the fifth step in Figure 7.38. From this sinusoidal acceleration of the board, the sinusoidal displacement of the board is calculated (the sixth step in Figure 7.38).

Then, the tensile stress of the outermost solder ball interconnections by PWB bending is calculated (the seventh step in Figure 7.38). The tensile stress is an input for calculating the number of cycles to 50% failure through the SN curve for solders (the eighth step in Figure 7.38). With this calculated number of cycles and the acceleration factor, vibration-induced solder ball fatigue reliability is predicted (the ninth step in Figure 7.38).

For the prediction of the board acceleration, the natural frequency and the transmissibility[24] are predicted by an FEA simulation. The first step in the development of an FEA model is to identify design features of the PWB, the BGA chip package and the solder ball interconnections. The design features are *PWB length*, *PWB width*, *PWB thickness*, *PWB material*, *BGA chip package length*, *BGA chip package width*, *BGA chip package thickness*, *length from the edge of PWB to the center of BGA chip package*, *BGA chip package material*, *solder ball height*, *solder ball diameter*, *solder ball pitch*, *the number of solder balls*, and *solder ball material*. These features are illustrated in Figure 7.40.

[24] Transmissibility is a term that is used to describe the response of a vibration isolation system. Literally, transmissibility is the ratio of displacement of an isolated system to the input displacement. It is used to describe the effectiveness of a vibration isolation system (http://www.herzan.com/herz3.htm).



where

H_{ball} : Solder ball height,

D_{ball} : Solder ball diameter,

$Pitch_{ball}$: Solder ball pitch,

L_{cp} : BGA chip package length,

W_{cp} : BGA chip package width,

Th_{cp}: BGA chip package thickness,

$D_{cp.c}$: Distance from the edge of PWB to the center of BGA,

L_{pwb}: PWB length,

W_{pwb}: PWB width,

Th_{pwb}: PWB thickness.

Figure 7.40 Solder ball interconnection design features for vibration-induced fatigue analysis

From the solution obtained for this FEA model, the natural frequency of the board is predicted. Then, with the natural frequency, the transmissibility is predicted from Equation (7.2) [Steinberg, 2001].

$$Q = \left(\frac{f_n}{G_{in}^{0.6}} \right)^{0.76} \tag{7.2}$$

where

Q : Transmissibility,

f_n : Natural frequency,

G_{in}: Input random acceleration (G).

With these data, the equivalent sinusoidal response of the board is calculated from Equation (7.3) [Steinberg, 2001].

$$G_{1.95\sigma} = 1.95 \times \sqrt{\frac{\pi}{2} G_{in} f_n Q} \tag{7.3}$$

where

$G_{1.95\sigma}$: Equivalent sinusoidal acceleration (G) ,

G_{in} : Input random acceleration (G),

Q : Transmissibility,

f_n : Natural frequency.

From this equivalent sinusoidal acceleration, board displacement is calculated. The relationship between sinusoidal acceleration and board displacement is shown in Equation (7.4) [Steinberg, 2001].

$$Y_{1.95\sigma} = \frac{G_{1.95\sigma} \times g}{(2\pi f_n)^2} \tag{7.4}$$

where

$Y_{1.95\sigma}$: Equivalent sinusoidal displacement,

$G_{1.95\sigma}$: Equivalent sinusoidal acceleration (G),

g : Acceleration of gravity,

f_n : Natural frequency.

Due to PWB bending displacement, solder balls experience cyclic stress. For predicting solder ball stress, another FEA model is developed. This FEA model is a one-dimensional model that uses the BEAM3 element type of ANSYS for the PWB; the COMBIN14 element type of ANSYS for solder balls; and the BEAM3 element type of

ANSYS for BGA chip package. The simulation results of this FEA model are illustrated in Figure 7.41. The boxes at the outermost solder balls experience maximum stress.

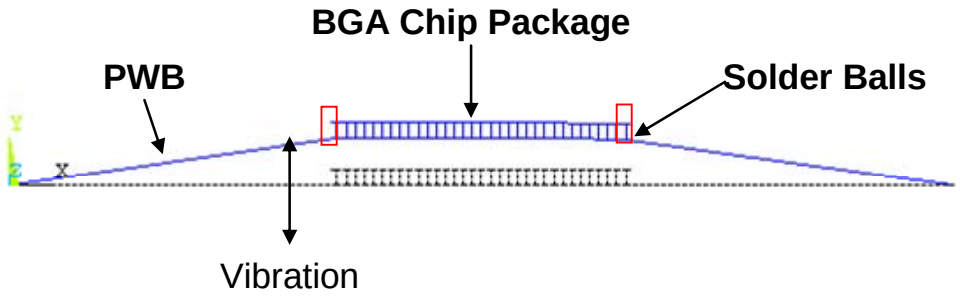


Figure 7.41 Simulation results of solder ball stress caused by PWB bending

The stress change of the outermost solder balls is a damage metric for predicting the number of cycles to 50% failure (N_{50}). For the calculation of the N_{50}, the SN curve for solder is used [Perkins and Sitaraman, 2004], shown in Table 7.22. The simulation results are also shown in Table 7.23. The simulation results are compared with those of Perkins, et al. [Perkins and Sitaraman, 2004], shown in Table 7.24. Since the natural frequency and the transmissibility results are in good agreement with Perkins's results, we think our FEA model is valid and applicable for system reliability prediction.

Table 7.22 SN curve for solder

$\sigma_a = \sigma'_f (2N_{50})^{C_{fde}}$	N_{50} : Number of cycles to 50% failure
	σ_a : Amplitude of stress change
	σ'_f : Fatigue ductility coefficient
	C_{fde} : Fatigue ductility exponent

Table 7.23 Test results of the vibration-induced fatigue model for solder ball interconnections

Natural Frequency	280
Transmissibility	38
Tensile Stress	4.18MPa
N_{50}	621600

Table 7.24 Validation of the vibration-induced fatigue model for solder ball interconnections

	Results of Perkins, et al.	Results of the FEA Model
Natural Frequency	314	280
Transmissibility	37	38

Total wearout failure reliability of solder ball interconnections (Figure 7.42) is assessed from both the thermo-mechanical fatigue model and the vibration-induced fatigue model following the superimposable failure mode structure and the algorithms defined in Chapter 5. Since the wearout failure reliability ($R_w(T_w)_{assessed}$ = 0.9930 at T_w= 90000 hours) is below target reliability ($R_w(T_w)_{target}$ =0.9950 at T_w= 90000 hours), design changes are necessary.

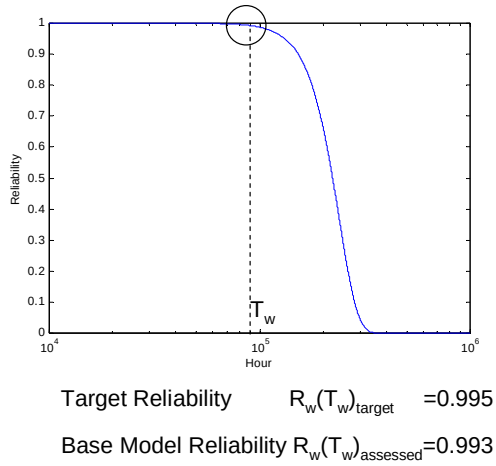


Figure 7.42 Total wearout failure reliability assessment results of the solder ball interconnections

7.4.4 Design Recommendations

Three design alternatives are recommended from the rules implemented in Table 5.15 with design constraints such as no-maintenance plan and fixed solder ball diameters. The first alternative is to add one more redundant component. These redundant systems are represented as a dashed polygon in Figure 7.43. The second alternative is to change the material properties of solder ball interconnections. New material properties are represented as a dashed box in Figure 7.44. The last alternative is to increase standoff height of solder balls. These design features are represented as a dashed box in Figure 7.45.

The resulting wearout reliability of the first alternative is $R_w(T_w)_{assessed}$=0.999 at T_w =90000 hours, and the wearout reliability of the second alternative is $R_w(T_w)_{assessed}$ =0.858 at T_w =90000. The wearout reliability of the last alternative is $R_w(T_w)_{assessed}$ =0.994 at T_w =90000, as shown in Figure 7.46. Since only the first option satisfies the target reliability, adding one redundant component is the selected recommended design change.

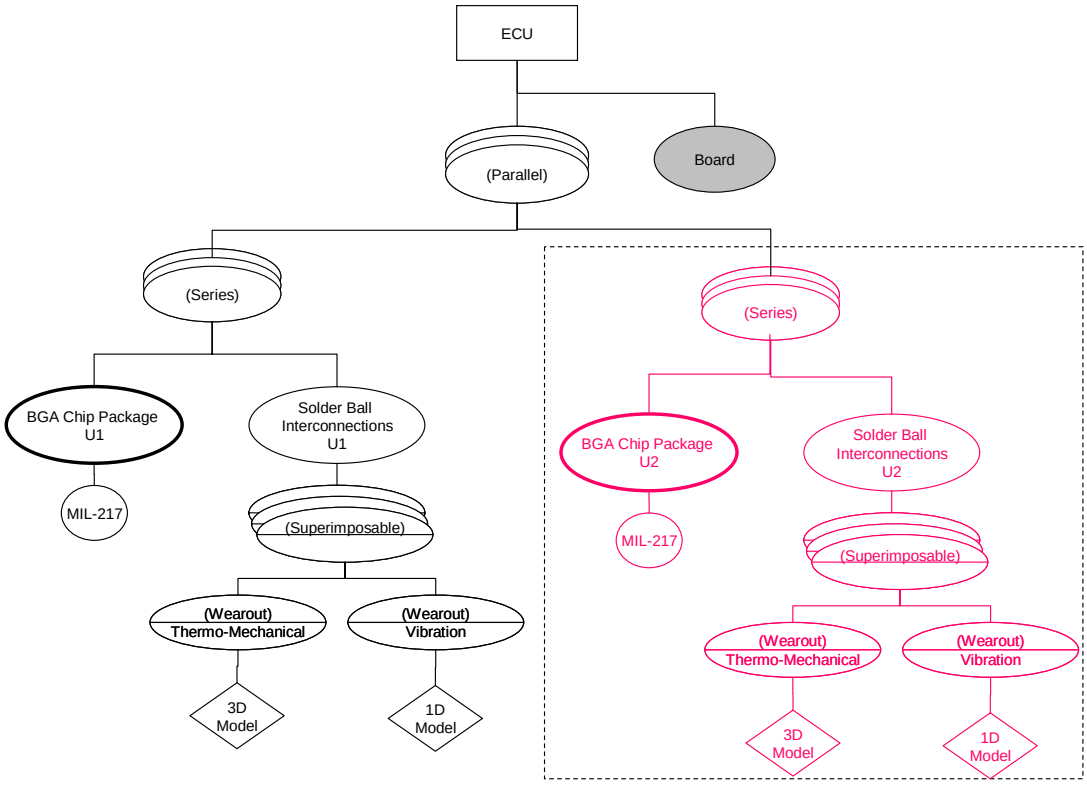


Figure 7.43 A ROM-Tree model for the design alternative 1

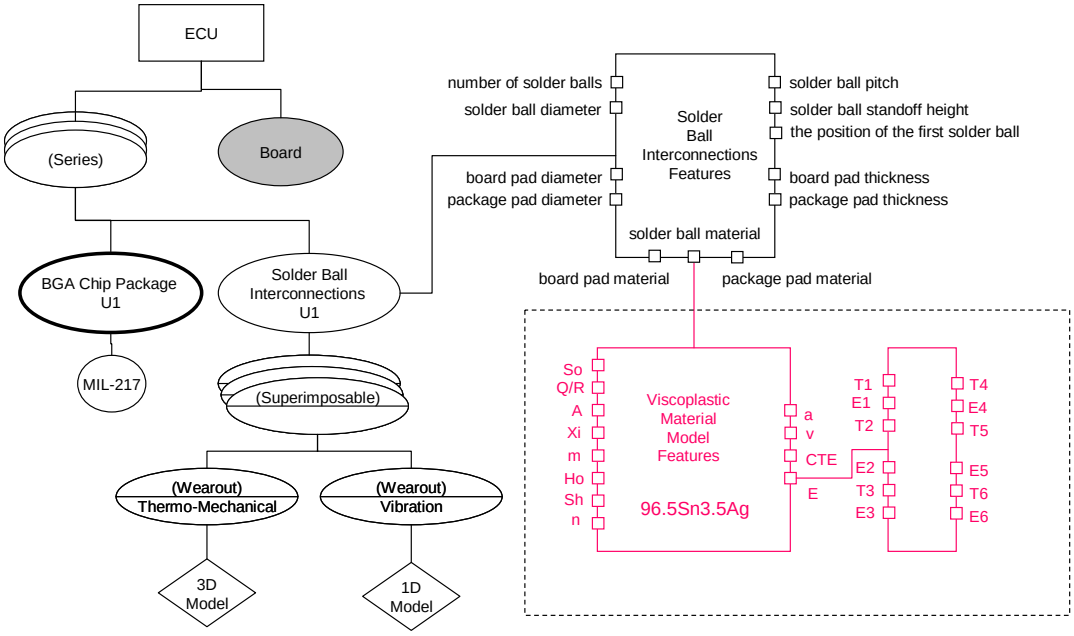


Figure 7.44 A ROM-Tree model for the design alternative 2

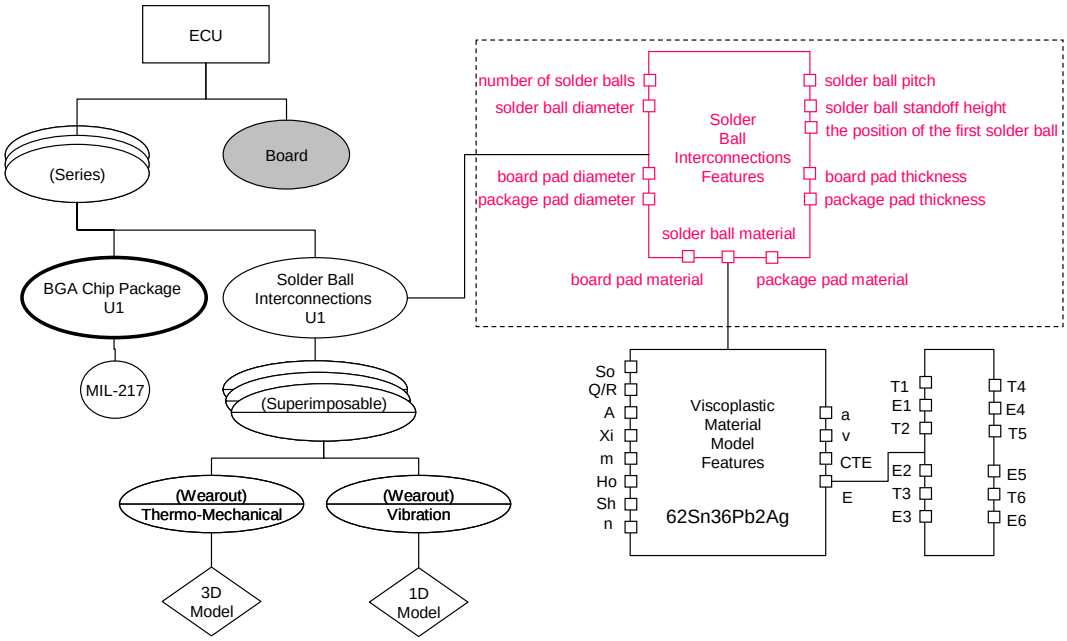


Figure 7.45 A ROM-Tree model for the design alternative 3

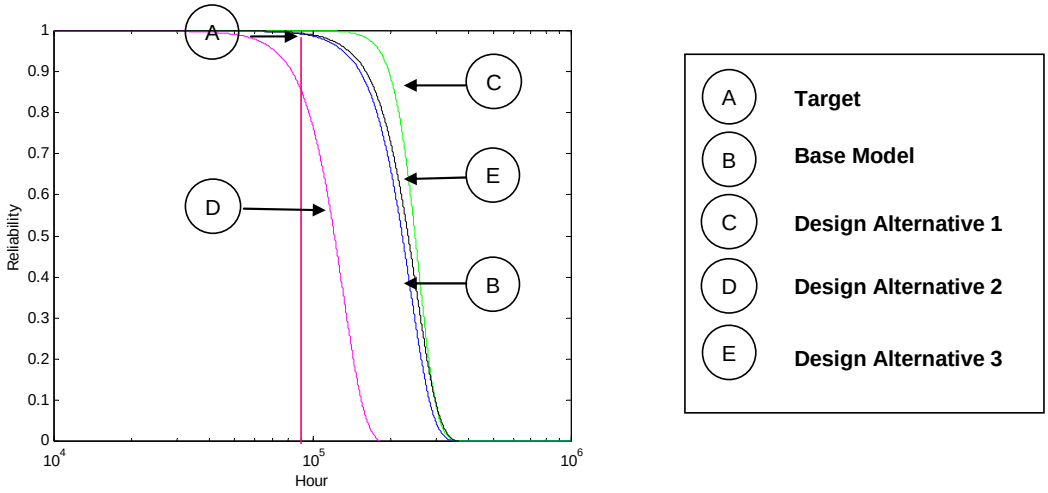
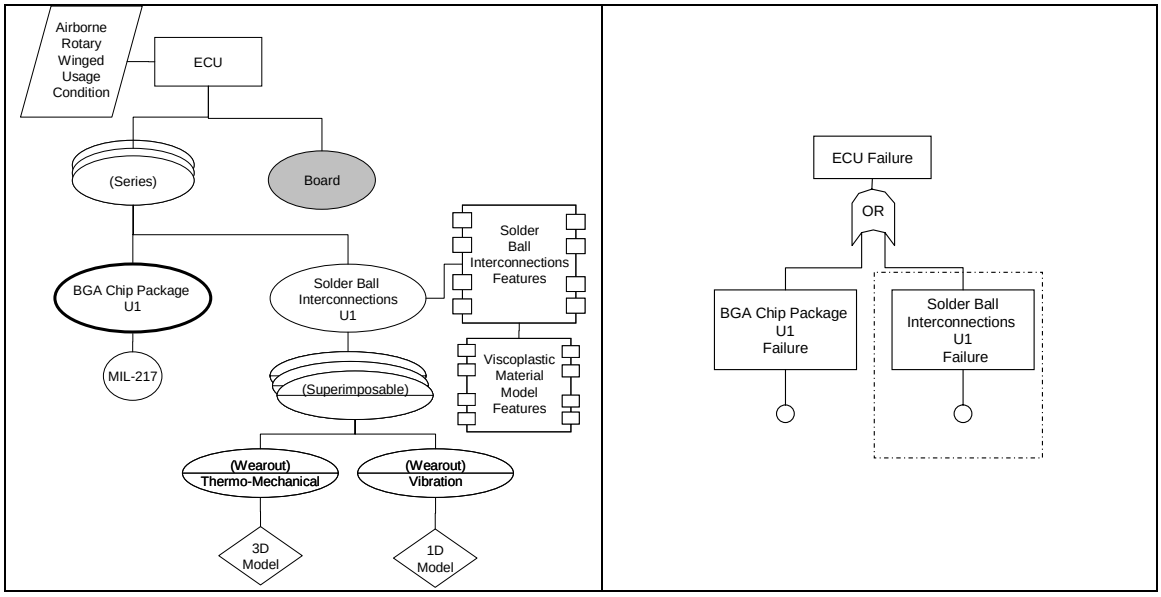


Figure 7.46 Wearout failure reliability assessment results of the design alternatives

7.4.5 Discussion



(a) ROM-Tree model (b) FTA model

Figure 7.47 Comparing ROM-Tree vs. FTA models for ECU design in harsh environments

The previous section demonstrates the ROM method for design recommendation. The ECU design in harsh environments is abbreviated in a ROM-Tree model in Figure 7.47-(a), and it is compared with an equivalent FTA model in Figure 7.47-(b). The ROM-Tree model in Figure 7.47-(a) shows a superimposable failure mode group for assessing complex failure modes in harsh environments and design features for design recommendations. However, the FTA model (Figure 7.47-(b)) is limited in such aspects. For example, the AND gate and the OR gate are not appropriate to represent the combined effect of thermo-mechanical failure and vibration failure because the FTA method is based on statistics principles, but not physics principles.

In addition to its qualitative effectiveness, shown in Figure 7.47, the ROM method is also quantitatively effective. For example, Table 7.25 provides a summary of the ECU design. According to Table 7.25, only design alternative 1 satisfies the target reliability. This shows that the ROM method makes quantity-based design changes possible.

Table 7.25 Summary of the ECU board assembly design

The ECU board assembly	$R_w(T_w)$ assessed (T_w = 90000 Hours)
Target reliability	0.995
Base design	0.993
Design alternative 1:Redundant components addition	0.999
Design alternative 2: Material change	0.858
Design alternative 3: Design feature change	0.994

The design recommendation for the ECU board assembly is also demonstrated through the prototype CASDfR tools. Figure 7.48 illustrates the recommended design alternatives in GT-SDfR-PWBA. Figure 7.49 shows models of design alternatives in GT-SDfR-PWBA.

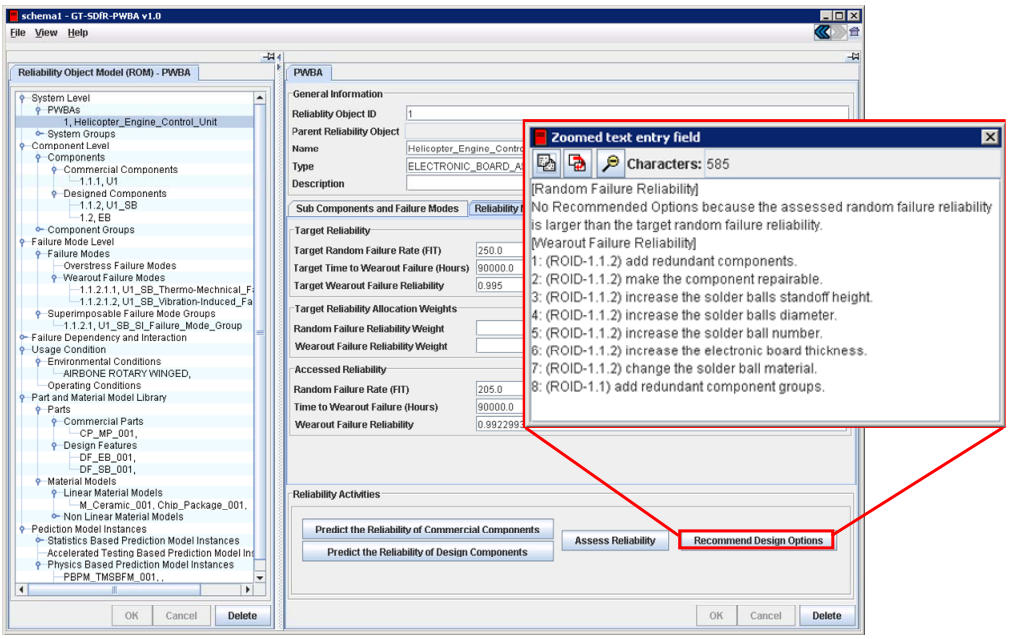


Figure 7.48 design recommendations for ECU board assembly implemented in GT-SDfR-PWBA

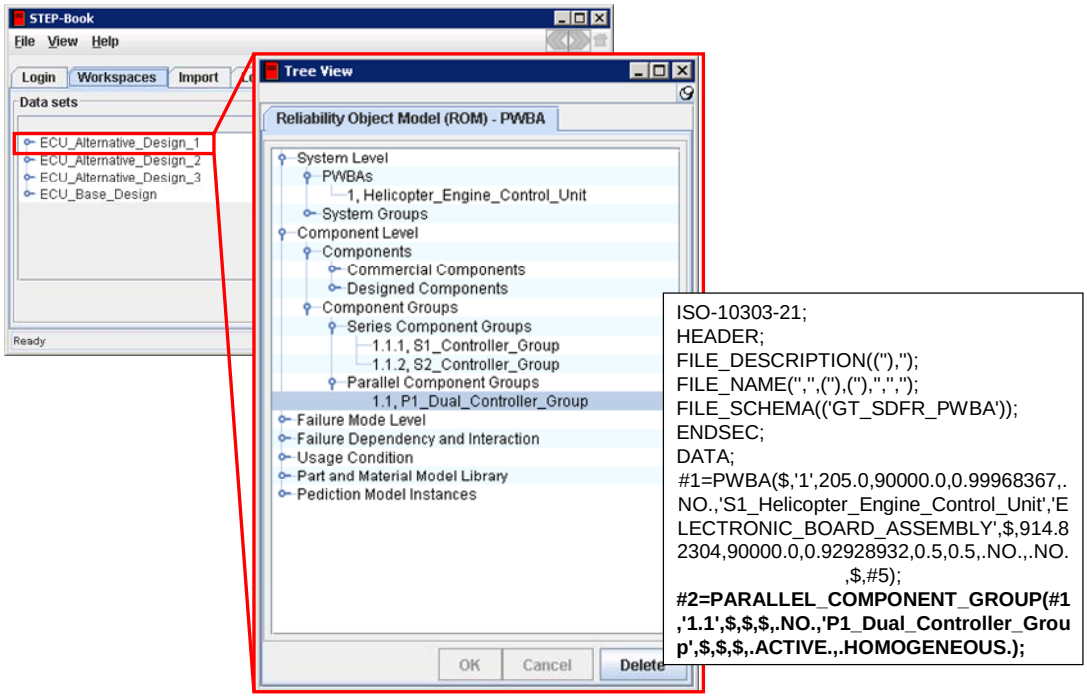


Figure 7.49 Models of design alternatives in GT-SDfR-PWBA

7.5 SUMMARY AND DISCUSSION

The four different case studies in the area of electronic systems demonstrate key aspects of the ROM method — reliability allocation, prediction, assessment, and design recommendations. In each test case, the wealth of design information provided by ROM is compared and contrasted with that by FTA to illustrate the advantages of our approach (Figures 7.8, 7.29, and 7.47). Besides, CASDfR-Framework and the prototype CASDfR tools are demonstrated through the test cases (Figures 7.9, 7.10, 7.30, 7.48, and 7.49).

One benefit of the ROM method that is not demonstrated in the presented case studied is the potential to achieve designing optimal system designs by concurrently evaluating all relevant design features and failure modes of components at the same system level. For example, an increase in board thickness increases the solder ball reliability against vibration failures, but it decreases PTH reliability against thermo-mechanical failures. Therefore, determining optimal board thickness requires a system reliability perspective that considers all such effects together.

ROM system reliability models for same case studies can be created that are more complex than the examples considered in this paper. For example, interactions that simultaneously involve many components — such as board warpage simulation, which represents the interaction between electronic components and a printed wiring board — are assumed to be negligible. However, the ROM method can accommodate such failure interactions by adding more failure prediction models or expert knowledge into each ROM model. As with almost any modeling technique, there is ultimately a trade-off between modeling fidelity and modeling cost.

CHAPTER 8

EVALUATION

This chapter describes the evaluation results of the ROM method with respect to the thesis objectives. First, Section 8.1 describes the evaluation scope and approach. Next, Section 8.2 analyzes the results obtained and evaluates them against the research objectives. Section 8.3 briefly summarizes the results of this evaluation.

8.1 EVALUATION SCOPE AND APPROACH

The primary goal of this research is to develop a knowledge model that facilitates the design of reliable electronic packaging systems in a systematic and timely manner and that incorporates all relevant aspects of system reliability knowledge. Reliability Object Model (ROM) has been developed to achieve this goal.

ROM consists of five components. They are ROM metrics, ROM algorithms, ROM-Structure, ROM-Library, and ROM-Tree. These ROM components as well as CASDfR-Framework are evaluated with respect to the research objectives.

Some developed parts in this study are beyond this evaluation scope because their main purposes are to demonstrate the ROM method. They are reliability prediction models and prototype CASDfR tools. Even if they are not evaluated in this chapter, they are already evaluated to a certain extent in the previous chapters.

The evaluation results contain what are developed for accomplishing the research objectives, how they are demonstrated, how they are novel or better than current

technologies, and what are limitations. Since most of them are already discussed in the previous chapters (Chapters 5, 6, and 7), associated parts are revisited or referenced here in summary form.

8.2 EVALUATION RESULTS

Table 8.1 shows the summary of the evaluation results with respect to the research objectives. The first column of this table contains the research objectives listed in Chapter 4. The next column group (*Test Cases*) contains the test cases developed in Chapter 7. The last column group (*ROM Components & CASDfR-Framework*) contains main components to be evaluated.

The results of the second column group show which test cases successfully demonstrate the ROM method to accomplish the objectives. A check mark (√) indicates a successful demonstration. A blank field indicates that the test case does not attempt to demonstrate the ROM method to accomplish the objectives.

The results of the third column group show which developed components accomplish the objectives successfully. A black dot (●) indicates a full accomplishment. A white dot (○) indicates a partial accomplishment. A blank field indicates that the component does not address the objective.

Detailed evaluation with respect to each objective follows the summary table.

Table 8.1 Summary of evaluation results with respect to objectives

Objectives (Pages 42-44)	Test Cases				Main Components					
	TC1	TC2	TC3	TC4	A	B	C	D	E	F
Objective 1: Unified Model	√	√	√	√			●	●	●	
Objective 2: Allocation Method	√	√				●	●		●	
Objective 3: Reliability Metrics	√	√	√	√	●		●			
Objective 4: Design Changes				√	●	●		○		
Objective 5: Implementation Framework		√	√	√						●

TC 1: the video broadcasting system test case,
TC 2: the notebook PC test case,
TC 3: the USB hub test case,
TC 4: the ECU test case.

√ : Successful demonstration,
Blank : Not covered.

A: ROM Metrics,
B: ROM Algorithms,
C: ROM-Structure,
D: ROM-Library,
E: ROM-Tree,
F: CASDfR-Framework.

● : Full accomplishment,
○ : Partial accomplishment,
Blank : Not addressed.

Objective 1: To develop a unified knowledge model for system design for reliability that can address both reliability allocation and assessment.

ROM includes a new unified reliability analysis structure that can address both reliability allocation and assessment consistently. This is achieved by three ROM components: ROM-Structure, ROM-Library, and ROM-Tree. ROM-Structure successfully represents four SDfR specifications: assembly structures (AS), logical structures (LS), failure mode structures (FMS), and failure interactions and dependencies (FID). ROM-Library successfully represents three SDfR specifications: design features (DF), reliability prediction models (RPM), and usage conditions (UC). ROM-Tree provides human-friendly graphical symbols of the unified reliability analysis structure.

Based on the unified reliability analysis structure, reliability allocation and assessment are demonstrated in Chapter 7. The ROM-Tree models for the video broadcasting system test case and the notebook PC test case (Figures 7.2, 7.6, and 7.7) are used for demonstrating reliability allocation. The ROM-Tree models for the USB hub test case and the ECU test case (Figures 7.13, 7.14, 7.15, and 7.32) are used for demonstrating reliability assessment.

The demonstration results show that the ROM method addresses both reliability allocation and assessment well. In addition, the results show that ROM is more effective for SDfR compared to existing approaches, providing richer semantics, unified methods, and improved SDfR quality (Table 5.23 and Figures 7.8 , 7.29, and 7.47). For example, ROM provides 55% more symbols than FTA (ROM: 17 symbols and FTA: 11 symbols).

The unified reliability analysis structure is developed mainly for electronic systems, but not for general systems. Therefore, it may be limited when it comes to

analyzing the reliability of systems in other areas such as complex network systems and biological systems.

Objective 2: To demonstrate that the developed unified knowledge model supports a representative target reliability allocation method for complex systems consisting of series and parallel subsystems.

The representative target reliability allocation method currently in ROM consists of three reliability weighting methods and five reliability allocation algorithms. The three reliability weighting methods are 1) a historical data-based relative target reliability weight method, 2) a design information-based relative target reliability weight method, and 3) a uniform relative target reliability weight method. The five reliability allocation algorithms are 1) a random failure reliability allocation algorithm for series structures, 2) a random failure reliability allocation algorithm for parallel structures, 3) a wearout failure reliability allocation algorithm for series structures, and 4) a wearout failure reliability allocation algorithm for parallel structures, and 5) a general reliability algorithm for complex structures.

As for weighting methods, the design information-based relative target reliability weight method and the uniform relative target reliability weight method are demonstrated by the video broadcasting system test case and the notebook PC test case (Tables 7.2 and 7.4). However, the historical data-based relative target reliability weight method is not demonstrated.

As for allocation algorithms, each reliability allocation algorithm is validated by a unit test case in Section 5.3.3 (Tables 5.4, 5.5, 5.6, 5.7, 5.8, and 5.9). Random failure

reliability allocation algorithms are demonstrated by the video broadcasting system test case (Table 7.1 and Figure 7.4). Wearout failure reliability allocation algorithms are demonstrated by the notebook PC test case (Table 7.5). The results show that ROM does a good job supporting the presented target reliability allocation method, and that it is useful for designing multi-level microelectronic systems for reliability.

Objective 3: To demonstrate that the developed unified knowledge model supports representative reliability metrics for random and wearout failures.

Two representative reliability metrics are presently defined in ROM. They are constant random failure rate (λ) for random failures and percentile wearout failure reliability ($R_w(T_w)$) for wearout failures.

These metrics are demonstrated for reliability allocation, prediction, and assessment and recommending design changes in Chapter 7. Target random failure rate and target wearout failure reliability are demonstrated by the video broadcasting system test case and the notebook PC test case. Assessed random failure rate and assessed wearout failure reliability are demonstrated by the USB hub test case and the ECU test case.

The demonstration results are compared with those of the conventional Part Count Method (Table 7.17). The Part Count Method assesses only random failure reliability (λ), and fatigue analysis methods assess only wearout failure reliability (N_{50}). However, the ROM method assesses both random and wearout failure reliability (λ and $R_w(T_w)$). Therefore, the ROM method provides more concise and complete reliability assessment results for design decisions. In addition, the ROM method makes design decisions for

reliability relatively easy because of the simplicity of the metrics. This is demonstrated by the ECU test case.

Since the ROM metrics are developed mainly for representing random and wearout failures of electronic systems, they are likely limited in their support for systems that require the consideration of infant and software failures.

Objective 4: To demonstrate that the developed unified knowledge model supports a representative method for recommending design changes.

The representative design recommendation method currently in ROM consists of sequential rule sets (Tables 5.14, 5.15, and 5.16). The first rule set is to determine if design changes are needed or not based on the difference between allocated and assessed reliabilities. The second rule set is to search for design objects that require changes. The third rule set is to recommend generic design alternatives based on statistics-based reliability knowledge in the general design domain. The fourth step is to recommend specific design alternatives based on physics-based reliability knowledge in specific design domains.

This method is demonstrated by the ECU test case in Section 7.4. The results show that the method makes quantity-based design changes possible and is useful for determining system configurations and design parameters (Table 7.25).

In this study, relatively simple design recommendation rules are developed and demonstrated because of the limitation of capturing diverse reliability knowledge. Implementing rules for estimating electromagnetic interferences (EMI) or limiting the

distance between high power chip packages will be necessary for complex electronics design support.

Objective 5: To implement the developed knowledge model in a computing framework, and to demonstrate the applicability of the framework using specific test cases and system-level reliability tools.

As for implementation of ROM, CASDfR-Framework and two prototype CASDfR tools (i.e. GT-SDfR-Manager and GT-SDfR-PWBA) have been developed in Chapter 6. CASDfR-Framework is a model-based information framework (Figure 6.4), and GT-SDfR-Manager and GT-SDfR-PWBA (Figures 6.7 and 6.8) are object-oriented technology-based system-level reliability tools.

As for demonstration of ROM, four electronic system test cases have been developed in Chapter 7. The video broadcasting system and the notebook PC design test cases demonstrates the reliability allocation perspective of the ROM method (Section 7.1 and Section 7.2). The USB hub design test case demonstrates the reliability prediction and assessment perspectives of the ROM method (Section 7.3). The ECU board design test case demonstrates the design recommendation perspective of the ROM method (Section 7.4).

The implementation and demonstration results show that ROM is computationally effective. This effectiveness may be further enhanced in the future by incorporating leading information technologies and standards such as XML, SysML, and STEP.

8.3 EVALUATION SUMMARY

This chapter evaluates the ROM method against the research objectives. The implementation presented in Chapter 6 and the results of the test cases presented in Chapter 7 are used as a basis for this evaluation.

This evaluation evidences the strengths and the limitations of ROM. Overall this evaluation indicates that the ROM method satisfactorily meets most of the research objectives. Therefore, we believe that the ROM method will increase the chance of developing reliable systems and decrease the burden of downstream reliability activities.

CHAPTER 9

CONTRIBUTIONS AND FUTURE WORK

Based on the research objectives, the developments, and the evaluations in the previous chapters, this chapter summarizes the research contributions and the opportunities for future work.

9.1 CONTRIBUTIONS

The primary contribution of this research is the development of a knowledge model method that facilitates designing reliable electronic packaging systems in a systematic and timely manner incorporating all relevant aspects of system reliability knowledge. The specific contributions with respect to the research objectives are summarized as follows:

Contribution 1: Development of a unified knowledge model for system design for reliability.

A new unified knowledge model for SDfR, referred to as Reliability Object Model (ROM), has been developed in Chapter 5. ROM consists of 1) a new unified reliability analysis structure (ROM-Tree) that can address both reliability allocation and assessment consistently, 2) representative reliability metrics that consider both random failures and wearout failures, 3) representative algorithms that allocate, predict, and assess reliability, and 4) representative rules for design change recommendation.

This reliability knowledge is captured in ROM in an efficient manner through three object-oriented construction aspects. The class structure of object-oriented schemes is used in constructing a new reliability analysis structure; the property definition of object-oriented schemes is used in defining reliability metrics; and the method definition of object-oriented schemes is used in defining reliability activities.

The effectiveness of ROM is demonstrated through representative capabilities for allocating, predicting, and assessing reliability and recommending design changes in Chapter 7. The results show that ROM is more effective for SDfR compared to existing methods, providing richer semantics, unified methods, and improved SDfR quality.

For Contributions 2-4, we claim ROM supports representative capabilities. By "representative" we mean that ROM can support similar algorithms and metrics like those demonstrated as long as the information content required by those alternate algorithms and metrics is a subset of the information content which ROM supports now, or which ROM can be extended to support from an implementation extension perspective (e.g., adding more attributes to existing ROM classes or adding more similar classes). However, we cannot claim ROM can support any arbitrary new algorithm or metric which would require research-oriented extensions to ROM's information content.

Contribution 2: Demonstration that ROM supports a representative target reliability allocation method for complex systems consisting of series and parallel subsystems.

An example target reliability allocation method for complex systems has been developed and described in Section 5.3.3. This method includes three reliability

weighting methods and five reliability allocation algorithms. The three reliability weighting methods are 1) a historical data-based relative target reliability weight method, 2) a design information-based relative target reliability weight method, and 3) a uniform relative target reliability weight method. The five reliability allocation algorithms are 1) random failure reliability allocation algorithm for series structures, 2) random failure reliability allocation algorithm for parallel structures, 3) wearout failure reliability allocation algorithm for series structures, and 4) wearout failure reliability allocation algorithm for parallel structures, and 5) general reliability algorithms for complex structures.

ROM support for this representative target reliability allocation method is evaluated in Section 5.3.3 and demonstrated in Sections 7.1 and 7.2. The results show that ROM successfully supports this allocation method in a manner that is useful for designing multi-level microelectronic systems for reliability.

Contribution 3: Demonstration that ROM supports representative effective reliability metrics for random and wearout failures.

Two independent reliability metrics are defined to account for random and wearout failures in Section 5.2. They are constant random failure rate (λ) for random failures and percentile wearout failure reliability ($R_w(T_w)$) for wearout failures. These metrics are demonstrated for reliability allocation, prediction, and assessment and recommending design changes in Chapter 7. The results show that ROM effectively supports these reliability metrics and that in doing so, ROM provides more complete information for system design compared to existing methods.

Contribution 4: Demonstration that ROM supports a representative method for recommending design changes.

A method for recommending design changes has been developed in Section 5.3.6. The method consists of four automated steps. The first step is to determine if design changes are needed or not based on the difference between allocated and assessed reliabilities. The second step is to search for design objects that require changes. The third step is to recommend generic design alternatives based on statistics-based reliability knowledge in the general design domain. The fourth step is to recommend specific design alternatives based on physics-based reliability knowledge in specific design domains.

This representative method starts with subsystem design changes and ends with top-level system design changes according to ROM-Structure. Since this method leads to design decisions at the subsystem level, component redundancies are more dominant rather than subsystem redundancies. Therefore, we believe that this method may be effective for reliable system design because systems that consist of component redundancies are more reliable than systems that consist of subsystem redundancies.

The design change recommendation method is demonstrated in Section 7.4. The results show that ROM successfully supports this recommendation method and thus makes quantity-based design changes possible in a way that is useful for determining system configurations and design parameters.

Contribution 5: Implementation of ROM in a computing framework, and demonstration of its applicability to the framework using specific test cases and prototype system-level reliability tools.

As for implementation of ROM, CASDfR-Framework and two prototype CASDfR tools (i.e. GT-SDfR-Manager and GT-SDfR-PWBA) have been developed in Chapter 6. CASDfR-Framework is a model-based information framework, and GT-SDfR-Manager and GT-SDfR-PWBA are object-oriented technology-based S/W tools. The implementation results show that ROM is successfully described explicitly in a computer interpretable form.

As for demonstration of ROM, four electronic system test cases have been developed in Chapter 7. The video broadcasting system and the notebook PC design test cases demonstrates the reliability allocation perspective of the ROM method. The USB hub design test case demonstrates the reliability prediction and assessment perspectives of ROM. The ECU board design test case demonstrates the design recommendation perspective of ROM. The demonstration exhibits the computational effectiveness of the ROM method as well as the possibility of realizing envisioned CASDfR.

9.2 FUTURE WORK

From the scope, the assumptions, and the test cases of this research in the previous chapters, we have identified several limitations and future extensions of the ROM method. These are summarized as follows:

Broad applications of the ROM method beyond electronic packaging systems. In this thesis, the ROM method is applied only to electronic packaging systems. However, we believe that the ROM method may be applied to more complicated application areas such as automobile, aerospace, and nuclear power plant applications.

Decomposition of ROM with respect to system functions and states. Although systems perform multiple functions and have multiple states, we do not specify system failures and reliability separately for each function and state in this research. Therefore, for more complete analysis of system reliability, ROM could be decomposed with respect to system functions and states.

Consideration of uncertainty factors during reliability prediction and assessment. Predicting system reliability includes many uncertainty factors such as material properties, usage conditions, and limitations in lifetime prediction models. Therefore, for a more accurate and reasonable reliability prediction and design decision, uncertainty factors could be considered in the SDfR context.

Incorporation of software reliability. Most complex electronic systems include not only hardware failures but also software failures. For example, when a cell phone fails, it may be caused by software bugs, hardware defects, or the combination of software and hardware faults. Therefore, the SDfR method might be extended so that it considers software reliability.

Extension of the ROM method for system lifecycle support. The ROM method is developed for the design stage. However, we believe that the ROM method could possibly be extended and used at other lifecycle stages such as manufacturing, maintaining, repairing, and planning next-version design stages. These extensions may be incorporated into existing product life cycle management (PLM) frameworks that are beginning to leverage new system-oriented standards such as SysML.

Extension of the ROM method considering cost. This research does not consider the cost aspect of design alternatives. If cost is considered, algorithms for optimal reliability allocation and design change recommendations may be possible.

Advanced reliability allocation methods. We have made some assumptions for weighting-based reliability allocation methods and found some limitations of them too. To develop advanced reliability allocation methods, we need to perform more research regarding aspects such as plausible cost models for optimization and effective heuristic methods that capture domain expert experience.

Quantitative compatibility comparison with other reliability analysis methods. If we have a ROM model for a given design, could we automatically process that ROM model and generate FTA, RBD, FMEA, and Markov chain models for that design? We think the answer is probably so. However, this needs to be determined by quantitative compatibility comparison at the meta-level among reliability analysis

methods. The comparison results might show the possibility of using ROM as a universal reliability analysis method.

APPENDIX A

TARGET RELIABILITY ALLOCATION ALGORITHMS

The following shows pseudo code for the target reliability allocation algorithms described in section 5.3.3.

Pseudo code conventions:

= :	represents assignment.
array_name [ ] :	represents an array. e.g. $ro_{i,j}[\,]$, and ***time*** [ ]
function_name ( ) :	represents a defined function. e.g. ***min()***, ***abs()***, ***power()***, and ***exp***()
(object)*.method_name()* :	represents a defined object method. e.g. (ro_i)*.get_sub-items()*
(object)*.attribute* :	represents a defined object attribute. e.g. ***(roi)****.target_random_failure_rate*
i :	represents a item level.
i.j :	represents a sub-item level.
Remark – :	represents comment statements.

A.1 Target Random Failure Reliability Allocation Algorithms

allocate_sub-item_target_random_failure_reliability *(* ro_i *)*
Remark – ro_i: the parent reliability object $_i$ (a system or system group).
Remark – The target reliability of ro_i is provided as an input to this algorithm.

Begin

Remark – get the sub-item list.
$ro_{i,j}[\,] = (ro_i)$*.get_sub-items( );*

Remark – Step 1: assign RTRWs of sub-items ($ro_{i,j}[\,]$).
assign_RTRWs_of_sub-items_for_random_failures($ro_{i,j}[\,]$*);*

Remark – Step 2: allocate sub-item target random failure reliability.
If (ro_i *is a system)* ***Then***
allocate_random_failure_reliability_for_series (ro_i, $ro_{i,j}[\,]$*);*
Else If (ro_i *is a series system group)* ***Then***
allocate_random_failure_reliability_for_series (ro_i, $ro_{i,j}[\,]$*);*
Else If (ro_i *is a parallel system group)* ***Then***
allocate_random_failure_reliability_for_parallel (ro_i, $ro_{i,j}[\,]$*);*
End If

Remark – Step 3: recursively call this same function.
For *(each* $ro_{i,j}$ *)*
If ($ro_{i,j}[j]$ *is not a simple homogeneous system)* ***Then***

allocate_sub-item_target_random_failure_reliability_reliability *(**ro**$_{i,j}$ [j]);*
End If
End For

End

allocate_random_failure_reliability_for_serie *(**ro**$_i$, **ro**$_{i,j}$ [ ])*
Remark – ***ro**$_i$*: the parent reliability object $_i$ (a system or system group).
Remark – ***ro**$_{i,j}$ [ ]*: the array of sub-items (systems or system groups).

Begin

Remark – Step 1: set the target random failure rate of the system.
*lambda$_i$ = (**ro**$_i$).target_random_failure_rate ;*

Remark – Step 2: allocate the target random failure rate of the sub-reliability objects.
For *(each **ro**$_{i,j}$)*
*w$_{i,j}$ = **ro**$_{i,j}$ [j].RTRW_for_random_failures ;*
__ro__$_{i,j}$ [j].target_random_failure_rate = lambda$_i$ × w$_{i,j}$;
End For

End

allocate_target_random_failure_reliability_for_parallel *(**ro**$_i$, **ro**$_{i,j}$ [ ])*
Remark – ***ro**$_i$*: the parent reliability object $_i$ (a system group).
Remark – ***ro**$_{i,j}$ [ ]*: the array of sub-items (systems or system groups).

Begin

Remark – set the variables of the system.
*lambda$_i$ = (**ro**$_i$).target_random_failure_rate ;*
*T$_w$ = (**ro**$_i$).target_time_to_wearout_failure ;*
*n = (**ro**$_i$).get_number_of_subsystems() ;*
*k = (**ro**$_i$).number_of_required_active_items ;*

Remark – set time segment for numerical calculation.
time_segment = 500 ;
time *[ ] = **from** 0 **to** T$_w$ by time_segment ;*

Remark – set the RTRWs of subsystems.
For *(each **ro**$_{i,j}$)*
*w$_{i,j}$ [j] = **ro**$_{i,j}$ [j].RTRW_for_random_failures ;*
End For

Remark – set the minimum weight.
*w$_{min}$ = **min** (w$_{i,j}$[ ]) ;*

Remark – Step 1: estimate the initial random failure rate of the minimum weight subsystem.
For *(each **time**)*
*x [t] = -1.0 × (10^-9) × **time** [t] ;*
*estimated_r$_{min}$ [t] = **power** ((**exp** ((-1.0 × lambda$_i$ × (10^-9) × **time** [t]))), w$_{min}$) ;*
*y [t] = **log** (estimated_r$_{min}$ [t]) ;*
End For

Remark – apply the Least Squares Method.
*lambda$_{min}$ = (y [ ] × **transpose** (x [ ])) / (x [ ] × **transpose** (x [ ])) ;*

error_ratio = 1.0 ;

While *(error_ratio > 0.001)*

Remark – Step 2: estimate the random failure rate of the system.
For *(each* $\mathbf{ro_{i,j}}$ *)*
$lambda_{i,j}[j] = lambda_{min} \times (w_{i,j} / w_{min})$;
End For

For *(each* ***time*** *)*
x [t] = -1.0 $\times$ *(10^-9)* $\times$ ***time*** *[t] ;*
$estimated\_r_{min}[t]$ = ***k-out-of-n_parallel_structure_equation*** $(lambda_{i,j}[\,], n, k)$;
y [t] = ***log*** *(*$estimated\_r_{min}[t]$*) ;*
End For

Remark – apply the least squares method.
$estimated\_lambda_i = (y[\,] \times$ ***transpose*** $(x[\,])) / (x[\,] \times$ ***transpose*** $(x[\,]))$;

Remark – Step 3: calculate the difference between the given target random failure rate and the estimated random failure rate.

$delta\_lambda_i = lambda_i - estimated\_lambda_i$;
error_ratio = ***abs*** $(delta\_lambda_i / lambda_i)$;

Remark – Step 4: estimate the next time-guess random failure rate of the minimum weight subsystem.
$lambda_{min} = lambda_{min} + delta\_lambda_i \times w_{min}$;

Remark – Step 5: repeate STEP 2 – STEP4 until the difference is smaller than the given tolerance.

End While

Remark – Step 6: calculate the allocated target random failure rates of the subsystems.
For *(each* $\mathbf{ro_{i,j}}$ *)*
$(\mathbf{ro_{i,j}}[j]).\ target\_random\_failure\_rate = lambda_{i,j}[j]$;
End For

End

A.2 Target Wearout Failure Reliability Allocation Algorithms

allocate_sub-item_target_wearout_failure_reliability (ro_i)
Remark – ***ro_i***: the parent reliability object $_i$ (a system or system group).
Remark – The target reliability of ***ro_i*** is provided as an input to this algorithm.

Begin

Remark – get the sub-item list
$ro_{i,j}$ [ ] = (ro_i).get_sub-items( );

Remark – Step 1: assign RTRWs of sub-items (***$ro_{i,j}$ [ ]***)
assign_RTRWs_of_sub-items_for_wearout_failures($ro_{i,j}$ [ ]);

Remark – Step 2: allocate sub-item target random failure reliability.
If (ro_i** is a system)* ***Then
    allocate_ wearout _failure_reliability_for_series (ro_i, $ro_{i,j}$ [ ]);
Else If (ro_i** is a series system group)* ***Then
    allocate_ wearout _failure_reliability_for_series (ro_i, $ro_{i,j}$ [ ]);
Else If (ro_i** is a parallel system group)* ***Then
    allocate_ wearout _failure_reliability_for_parallel (ro_i, $ro_{i,j}$ [ ]);
End If

Remark – Step 3: recursively call this same function.
***For** (each **$ro_{i,j}$**)*
  If ($ro_{i,j}$ [j]** is not a simple homogeneous system)* ***Then
    allocate_sub-item_target_ wearout_failure_reliability_reliability ($ro_{i,j}$ [j]);
  End If
End For

End

allocate_target_wearout_failure_reliability_for_series (ro_i, $ro_{i,j}$ [ ])
Remark – ***ro_i***: the parent reliability object $_i$ (a system or system group).
Remark – ***$ro_{i,j}$ [ ]***: the array of sub-items (systems or system groups).

Begin

Remark – Step 1: set the target wearout failure reliability of the system.
r_i = ***(ro_i).target_wearout_failure_reliability ;***

Remark – Step 2: allocate the target wearout failure reliabilities of the subsystems.
***For** (each **$ro_{i,j}$**)*
    $w_{i,j}$ = ***$ro_{i,j}$ [j].RTRW_for_wearout_failures ;***
    $ro_{i,j}$ [j].target_wearout_failure_reliability = power (r_i, $w_{i,j}$);
End For

End

allocate_target_wearout_failure_reliability_for_parallel *(**ro**$_i$, **ro**$_{i,j}$ [ ])*
Remark – ***ro***$_i$: the parent reliability object $_i$ (a system group).
Remark – ***ro***$_{i,j}$ *[ ]*: the array of sub-items (systems or system groups).

Begin

Remark – set the variables of the system.
r_i = (***ro***$_i$).target_wearout_failure_reliability ;
n = (***ro***$_i$).get_number_of_subsystems() ;
k = (***ro***$_i$).number_of_required_active_items ;

Remark – set the RTRWs of the subsystems.
For *(each **ro**$_{i,j}$)*
$w_{i,j}$ = (***ro***$_{i,j}$ *[j]).RTRW_for_wearout_failures ;*
End For

Remark – set the minimum weight
w_{min} = ***min*** ($w_{i,j}$ [ ]) ;

Remark – STEP 1: estimate the initial wearout failure reliability of the minimum weight subsystem.
estimated_r_{min} = ***power*** (r_i , w_{min}) ;
error_ratio = 1.0 ;

While *(error_ratio > 0.001)*

Remark – STEP 2: estimate the wearout failure reliability of the system.
For *(each **ro**$_{i,j}$)*
$r_{i,j}$ *[j]* = ***power*** (estimated_r_{min} , $w_{i,j}$ / w_{min}) ;
End For

estimated_r_i = ***k-out-of-n_parallel_structure_equation*** ($r_{i,j}$ [j], n, k) ;

Remark – STEP3 : calculate the difference between the given target wearout failure reliability and the estimated wearout failure reliability.
delta_r_i = r_i - estimated_r_i ;
error_ratio = ***abs*** (delta_r_i / r_i) ;

Remark – STEP4: estimate the next time-guess wearout failure reliability of the minimum weight subsystem.
estimated_r_{min} = estimated_r_{min} + (1- ***power***(1-delta_r_i, w_{min}));

Remark – STEP 5: repeat STEP 2 – STEP4 until the difference is smaller than the given tolerance.

End While

Remark – STEP 6: calculate the allocated target wearout failure reliabilities of the subsystems.

For *(each **ro**$_{i,j}$)*
(***ro***$_{i,j}$ *[j]).* target_wearout_failure_reliability = $r_{i,j}$ [j] ;
End For

End

APPENDIX B

RELIABILITY ASSESSMENT ALGORITHMS

The following shows pseudo code for target reliability assessment algorithms described in section 5.3.5.

Pseudo code conventions:

= :	represents assignment.
array_name [ ] :	represents an array. e.g. $ro_{i,j}[\,]$, and ***time*** [ ]
function_name ( ) :	represents a defined function. e.g. ***min()***, ***abs()***, ***power()***, and ***exp***()
(object)*.method_name()* :	represents a defined object method. e.g. (ro_i)*.get_sub-items()*
(object)*.attribute* :	represents a defined object attribute. e.g. ***(roi)****.target_random_failure_rate*
i :	represents a item level.
i.j :	represents a sub-item level.
Remark – :	represents comment statements.

B.1 Random Failure Reliability Assessment Algorithms

assess_item_random_faiure_reliability_from_sub-item_reliabilities *(* ro_i *)*
Remark – ro_i: reliability object $_i$ (a system, system group, or component group).

Begin

$ro_{i,j}[\,] = (ro_i)$*.get_sub-items( );*

Remark – Step 1: recursively call this same function.
For *(each* $ro_{i,j}$ *)*
  If *((*$ro_{i,j}[j]$*).get_sub-items exist)* ***Then***
    assess_item_random_faiure_reliability_from_sub-item_reliabilities *(*$ro_{i,j}[j]$*);*
  End If
End For

Remark – Step 2: assess item reliability from sub-item reliabilities.
If *(*ro_i *is based on a series structure)* ***Then***
  assess_ random_faiure_reliability_for_series *(*ro_i, $ro_{i,j}[\,]$*);*
Else If(*ro_i *is based on a parallel structure)* ***Then***
  assess_ random_faiure_reliability_for_parallel *(*ro_i, $ro_{i,j}[\,]$*);*
End If

End

assess_random_failure_reliability_for_series (***ro_i, $ro_{i,j}$ [ ]***)
Remark – ***ro_i***: the parent reliability object $_i$ (a system, system group, or component group).
Remark – ***$ro_{i,j}$ [ ]***: the array of sub-items (systems, systems groups, components, or component groups).

Begin

Remark – Step 1: initialize the random failure rate of the reliability object.
*(**ro_i**).random_failure_rate = 0 ;*

Remark – Step 2: assess the random failure rate of the reliability object.
For *(each **$ro_{i,j}$**)*
    *(**ro_i**).random_failure_rate = (**ro_i**).random_failure_rate + (**$ro_{i,j}$** [j]).random_failure_rate ;*
End For

End

assess_random_failure_reliability_for_parallel (***ro_i, $ro_{i,j}$ [ ]***)
Remark – ***ro_i***: the parent reliability object $_i$ (a system group or component group).
Remark – ***$ro_{i,j}$ [ ]***: the array of sub-items (systems, systems groups, components, or component groups).

Begin

Remark – Step 1: set the variables of the reliability object.
*(**ro_i**).random_failure_rate = 0 ;*
*T_w =(**ro_i**).target_time_to_wearout_failure ;*
*n = (**ro_i**).get_number_of_subsystems() ;*
*k = (**ro_i**).number_of_required_active_items ;*

Remark – Step 2: set time segment for numerical calculation.
time_segment = 500 ;
time *[ ] = **from** 0 **to** T_w by time_segment ;*

Remark – Step 3: set the random failure rates of the sub-items.
For *(each **$ro_{i,j}$**)*
    *$lambda_{i,j}$ [j] = (**$ro_{i,j}$** [j]).random_failure_rate ;*
End For

Remark – Step 4: calculate random failure reliability values of the reliability object.
For *(each **time**)*
    *x [t] = -1.0 × (10^-9) × **time** [t] ;*
    *assessed_r [t] = **k-out-of-n_parallel_structure_equation** ($lambda_{i,j}$ [ ], n, k) ;*
    *y [t] = **log** (assessed_r [t]) ;*
End For

Remark – Step 5: apply the least squares method and assess the random failure rate of the reliability object.
*(**ro_i**).random_failure_rate = (y [ ] × **transpose** (x [ ])) / (x [ ] × **transpose** (x [ ])) ;*

End

B.2 Wearout Failure Reliability Assessment Algorithms

assess_item_wearout_faiure_reliability_from_sub-item_reliabilities (ro_i)
Remark – ***ro_i***: reliability object $_i$ (a system, system group, component, or component group).

Begin

$ro_{i,j}$ [ ] = (ro_i).get_sub-items( );

Remark – Step 1: recursively call this same function.
For *(each* ***$ro_{i,j}$*** *)*
    If *(($ro_{i,j}$ [j]).get_sub-items exist)* ***Then***
        assess_item_wearout_faiure_reliability_from_sub-item_reliabilities *($ro_{i,j}$ [j]);*
    End If
End For

Remark – Step 2: assess item reliability from sub-item reliabilities
If *(****ro_i*** *is based on a series structure)* ***Then***
    assess_wearout_faiure_reliability_for_series *(****ro_i****, ****$ro_{i,j}$*** *[ ]);*
Else If(*ro_i*** *is based on a parallel structure)* ***Then***
    assess_wearout_faiure_reliability_for_parallel *(****ro_i****, ****$ro_{i,j}$*** *[ ]);*
Else If *(****ro_i*** *is based on an independent failure model structure)* ***Then***
    assess_wearout_faiure_reliability_for_independent *(****ro_i****, ****$ro_{i,j}$*** *[ ]);*
Else If(*ro_i*** *is based on a superimposable failure model structure)* ***Then***
    assess_wearout_faiure_reliability_for_superimposable *(****ro_i****, ****$ro_{i,j}$*** *[ ]);*
End If

End

assess_wearout_faiure_reliability_for_series *(****ro_i****, ****$ro_{i,j}$*** *[ ])*
Remark – ***ro_i***: the parent reliability object $_i$ (a system, system group, or component group).
Remark – ***$ro_{i,j}$*** *[ ]*: the array of sub-items (systems, system groups, components, or component groups).

Begin

Remark – Step 1: initialize the wearout failure reliability of the reliability object
*(****ro_i****).wearout_failure_reliability = 1 ;*

Remark – Step 1: assess the wearout failure reliability of the reliability object
For **(** *each* ***$ro_{i,j}$*** **)**
    *(****ro_i****).wearout_failure_reliability = (****ro_i****).wearout_failure_reliability*
        ****** *(****$ro_{i,j}$*** *[j]).wearout_failure_reliability ;*
End For

End

assess_wearout_faiure_reliability_for_parallel *(**ro**$_i$, **ro**$_{i,j}$ [ ])*
Remark – ***ro**$_i$*: the parent reliability object $_i$ (a system group or component group).
Remark – ***ro**$_{i,j}$ [ ]*: the array of sub-items (systems, system groups, components, or component groups).

Begin

Remark – Step 1: set the variables of the reliability object
*(**ro**$_i$).wearout_failure_reliability = 1 ;*
*n = (**ro**$_i$).get_number_of_subsystems() ;*
*k = (**ro**$_i$).number_of_required_active_items ;*

Remark – Step 2: get the wearout failure reliabilities of the sub-items

For (*each* ***ro**$_{i,j}$*)
*wearout_failure_reliability $_{i,j}$ [j] = (**ro**$_{i,j}$ [j]). wearout_failure_reliability ;*
End For

Remark – Step 3: assess the wearout failure reliability of the reliability object
*(**ro**$_i$).wearout_failure_reliability =* ***k-out-of-n_parallel_structure_equation*** *(*
wearout_failure_reliability $_{i,j}$ [ ], n, k) ;

End

assess_wearout_faiure_reliability_for_independent *(**ro**$_i$, **ro**$_{i,j}$ [ ])*
Remark – ***ro**$_i$*: the parent reliability object $_i$ (a component or failure mode group).
Remark – ***ro**$_{i,j}$ [ ]*: the array of sub-items (failure modes or failure mode groups).

Begin

Remark – Step 1: initialize the wearout failure reliability of the reliability object.
*(**ro**$_i$).wearout_failure_reliability = 1 ;*

Remark – Step 2: assess the wearout failure reliability of the reliability object.
For (*each* ***ro**$_{i,j}$*)
*(**ro**$_i$).wearout_failure_reliability = (**ro**$_i$).wearout_failure_reliability*
** (**ro**$_{i,j}$ [j]).wearout_failure_reliability ;*
End For

End

assess_wearout_faiure_reliability_for_superimposable *(**ro**$_i$, **ro**$_{i,j}$ [ ])*
Remark – ***ro**$_i$*: the parent reliability object $_i$ (a failure mode group).
Remark – ***ro**$_{i,j}$ [ ]*: the array of sub-items (failure modes).

Begin

Remark – Step 1: set the number of sampling points for cycles to n percent failures.
sp = 3 ;

Remark – Step 2: set sampling cycles to n percent failures (e.g., $N_{25\%}$, $N_{50\%}$, and $N_{75\%}$)
of each failure mode.
For *(each* ***ro**$_{i,j}$*)*
For *(y = 1: 1: sp)*
Remark – the number of cycles to (y/(sp+1))*100 percent failures
*$N_{\_npf}$[j,y] = (**ro**$_{i,j}$ [j]).alpha*

```
                    * ( (-1.0*log(1-y/( sp+1)))^(1.0/ (ro_i,j [j]).beta) )
                    * (ro_i,j [j]).frequency;
    End For
End For
```

Remark – Step 3: calculate cycles to n percent failures of the component using Miner's Rule.

```
For (y = 1: 1: sp)
    sum[y]   = 0;
    For (each ro_i,j)
        Remark –  calculate total accumlative damage per unit time (∑ n_i/N_i)
        sum [y]  = sum [y] + (ro_i,j [j]).frequency /N_npf[x,y];
    End For
      Remark – calculate each time-to-x-percent-failure (T_npf(y)).
      T_npf(y) = 1/ sum [y] ;

End For
```

Remark – Step 4: alculate alpha and beta parameters for the Weibull function of the component.
apply the least squares method (AX=B, $A^TAX=A^TB$, $X = (A^TA)^{-1}A^TB$).

```
For (y = 1: 1: sp)
    A[y,1] = log(-log(1-y/( sp+1)));
    A[y,2] = 1;
    B[y] = log(T_npf(y));
End For
X = (A^TA)^-1 A^TB ;
```

Remark – set alpha and beta parameters for the Weibull function of the component.

```
alpha = exp(X[2]) ;
beta  = 1/X[1] ;
```

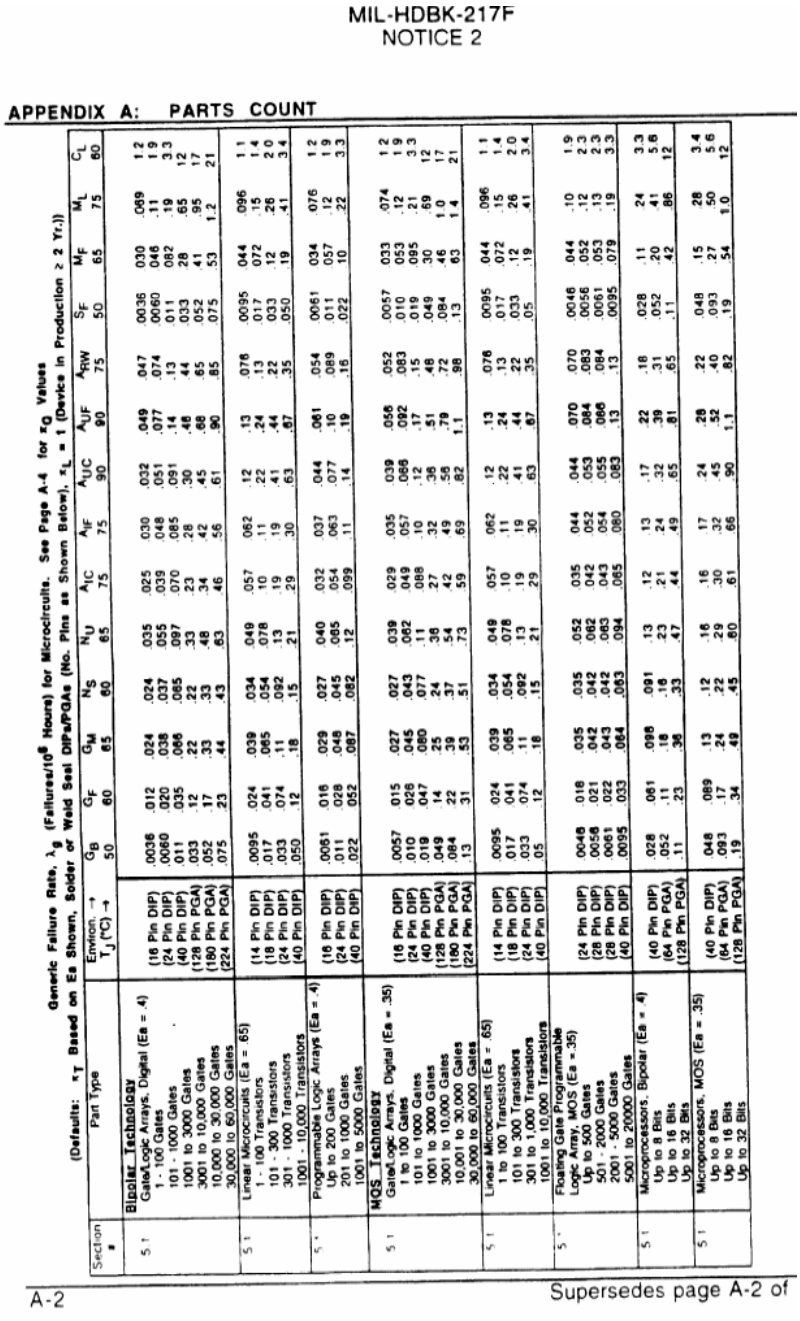
Remark – Step 5: assess the wearout failure reliability of the reliability object.

```
(ro_i).wearout_failure_reliability =  exp(-((ro_i).target_time_to_wearout_failure/alpha)^beta);

End
```

APPENDIX C

RELIABILITY PREDICTION MODELS

C.1 An Example Page of MIL-HDBK-217



MIL-HDBK-217F
NOTICE 2

APPENDIX A: PARTS COUNT

Generic Failure Rate, λ_g (Failures/10^6 Hours) for Microcircuits. See Page A-4 for π_Q Values
(Defaults: π_T Based on Ea Shown, Solder or Weld Seal DIPs/PGAs (No. Pins as Shown Below), π_L = 1 (Device in Production ≥ 2 Yr.))

Section #	Part Type	Environ. → / T_J (°C) →	G_B 50	G_F 60	G_M 65	N_S 60	N_U 65	A_{IC} 75	A_{MF} 75	A_{UC} 90	A_{UF} 90	A_{RW} 75	S_F 50	M_F 65	M_L 75	C_L 60
	Bipolar Technology															
5.1	Gate/Logic Arrays, Digital (Ea = .4)															
	1 - 100 Gates	(16 Pin DIP)	.0036	.012	.024	.024	.035	.025	.030	.032	.049	.047	.0036	.030	.069	1.2
	101 - 1000 Gates	(24 Pin DIP)	.0060	.020	.038	.037	.055	.039	.048	.051	.077	.074	.0060	.046	.11	1.9
	1001 to 3000 Gates	(40 Pin DIP)	.011	.035	.066	.065	.097	.070	.085	.091	.14	.13	.011	.082	.19	3.3
	3001 to 10,000 Gates	(128 Pin PGA)	.033	.12	.22	.22	.33	.23	.28	.30	.45	.44	.033	.28	.65	1.2
	10,001 to 30,000 Gates	(180 Pin PGA)	.052	.17	.33	.33	.48	.34	.42	.45	.68	.65	.052	.41	.95	1.7
	30,000 to 60,000 Gates	(224 Pin PGA)	.075	.23	.44	.43	.63	.46	.56	.61	.90	.85	.075	.53	1.2	2.1
5.1	Linear Microcircuits (Ea = .65)															
	1 - 100 Transistors	(14 Pin DIP)	.0095	.024	.039	.034	.049	.057	.062	.12	.13	.076	.0095	.044	.096	1.1
	101 - 300 Transistors	(18 Pin DIP)	.017	.041	.065	.054	.078	.10	.11	.22	.24	.13	.017	.072	.15	1.4
	301 - 1000 Transistors	(24 Pin DIP)	.033	.074	.11	.092	.13	.19	.19	.41	.44	.22	.033	.12	.26	2.0
	1001 - 10,000 Transistors	(40 Pin DIP)	.050	.12	.18	.15	.21	.29	.30	.63	.67	.35	.050	.18	.41	3.4
5.1	Programmable Logic Arrays (Ea = .4)															
	Up to 200 Gates	(16 Pin DIP)	.0061	.016	.029	.027	.040	.032	.037	.044	.061	.054	.0061	.034	.076	1.2
	201 to 1000 Gates	(24 Pin DIP)	.011	.028	.048	.045	.065	.054	.063	.077	.10	.089	.011	.057	.12	1.9
	1001 to 5000 Gates	(40 Pin DIP)	.022	.052	.087	.082	.12	.099	.11	.14	.19	.16	.022	.10	.22	3.3
	MOS Technology															
5.1	Gate/Logic Arrays, Digital (Ea = .35)															
	1 to 100 Gates	(16 Pin DIP)	.0057	.015	.027	.027	.039	.029	.035	.039	.056	.052	.0057	.033	.074	1.2
	101 to 1000 Gates	(24 Pin DIP)	.010	.026	.045	.043	.062	.049	.057	.066	.092	.083	.010	.053	.12	1.9
	1001 to 3000 Gates	(40 Pin DIP)	.018	.047	.080	.077	.11	.088	.10	.12	.17	.15	.019	.095	.21	3.3
	3001 to 10,000 Gates	(128 Pin PGA)	.049	.14	.25	.24	.36	.27	.32	.36	.51	.48	.049	.30	.69	1.2
	10,001 to 30,000 Gates	(160 Pin PGA)	.084	.22	.39	.37	.54	.42	.49	.58	.79	.72	.084	.46	1.0	1.7
	30,000 to 60,000 Gates	(224 Pin PGA)	.13	.31	.53	.51	.73	.59	.69	.82	1.1	.98	.13	.63	1.4	2.1
5.1	Linear Microcircuits (Ea = .65)															
	1 to 100 Transistors	(14 Pin DIP)	.0095	.024	.039	.034	.049	.057	.062	.12	.13	.076	.0095	.044	.096	1.1
	101 to 300 Transistors	(18 Pin DIP)	.017	.041	.065	.054	.078	.10	.11	.22	.24	.13	.017	.072	.15	1.4
	301 to 1,000 Transistors	(24 Pin DIP)	.033	.074	.11	.092	.13	.19	.19	.41	.44	.22	.033	.12	.26	2.0
	1001 to 10,000 Transistors	(40 Pin DIP)	.05	.12	.18	.15	.21	.29	.30	.63	.67	.35	.05	.19	.41	3.4
5.1	Floating Gate Programmable															
	Logic Array, MOS (Ea = .35)															
	Up to 500 Gates	(24 Pin DIP)	.0048	.018	.035	.035	.052	.035	.044	.044	.070	.070	.0046	.044	.10	1.9
	501 - 2000 Gates	(28 Pin DIP)	.0058	.021	.042	.042	.062	.042	.052	.053	.084	.083	.0056	.052	.12	2.3
	2001 - 5000 Gates	(28 Pin DIP)	.0061	.022	.043	.042	.063	.043	.054	.055	.086	.084	.0061	.053	.13	2.3
	5001 to 20000 Gates	(40 Pin DIP)	.0095	.033	.064	.063	.094	.065	.080	.083	.13	.13	.0095	.079	.19	3.3
5.1	Microprocessors, Bipolar (Ea = .4)															
	Up to 8 Bits	(40 Pin DIP)	.028	.061	.098	.091	.13	.12	.13	.17	.22	.18	.028	.11	.24	3.3
	Up to 16 Bits	(64 Pin PGA)	.052	.11	.18	.16	.23	.21	.24	.32	.39	.31	.052	.20	.41	5.6
	Up to 32 Bits	(128 Pin PGA)	.11	.23	.38	.33	.47	.44	.49	.65	.81	.65	.11	.42	.86	12
5.1	Microprocessors, MOS (Ea = .35)															
	Up to 8 Bits	(40 Pin DIP)	.048	.089	.13	.12	.16	.16	.17	.24	.28	.22	.048	.15	.28	3.4
	Up to 16 Bits	(64 Pin PGA)	.093	.17	.24	.22	.29	.30	.32	.45	.52	.40	.093	.27	.50	5.6
	Up to 32 Bits	(128 Pin PGA)	.19	.34	.49	.45	.60	.61	.66	.90	1.1	.82	.19	.54	1.0	12

A-2

Supersedes page A-2 of

C.2 A Thermo-Mechanical Fatigue Analysis Model for PTHs

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!The Two Dimensional Numerical Fatigue Analysis Model
!!of Plated Through Hole (Plane Strain Model)

!!by Injoong Kim
!!12/5/05

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 1. startup

finish
/clear,nostart
/title, Plated Through Hole Reliability

/RGB,INDEX,100,100,100, 0
/RGB,INDEX, 80, 80, 80,13
/RGB,INDEX, 60, 60, 60,14
/RGB,INDEX, 0, 0, 0,15
/REPLOT

!Units are mm,N/mm^2 (same as MPa) etc.
!Units

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2. preprocessor

/prep7

!! preference
KEYW,PR_SET,1
KEYW,PR_STRUC,1
KEYW,PR_THERM,1
KEYW,PR_MULTI,1

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.1 define parameters

!! Constants
incc = 1e-6      !tiny increment for selecting objects
n = 1            !mesh conversion factor 1/2/3/4

!! Material Index
pt_hole=1        !copper
board=2          !FR4

!! Geometric Parameters
!D_H             !Hole Diameter (mm)
!TH_P            !Plating Thickness (mm)
!D_PAD           !Pad Diameter (mm)
!TH_PAD          !Pad Thickness (mm)
!TH_B            !Board Thickness (mm)

!! Operating Parameters
!T_0             !Ambient Temperature (deg K)
!T_C0            !Steady State Temperature of Component (PTH) (deg K)
```

```
!T_C1          !State 1 Temperature of Component (PTH) (deg K)
!T_C2          !State 2 Temperature of Component (PTH) (deg K)
!T_B0          !Steady State Temperature of of PWB (deg K)
!T_B1          !State 1 Temperature of PWB (deg K)
!T_B2          !State 2 Temperature of PWB (deg K)
!time_RMP      !Ramp hour (Hour)
!time_DHI      !Duration hour at high temperature  (Hour)
!time_DLO      !Duration hour at low temperature   (Hour)

!! Material Parameters

!! pt_hole
!E_C           !Young's Modulus of Component (PTH)(MPa)
!V_C           !Poisson's Ratio of Component (PTH)
!G_C           !Shear Modulus of Component (PTH) (MPa)
!CTE_C         !CTE of Component (PTH) (mm/mmK)
!STRA_C_1      !Strain of Component (PTH)
!STRA_C_2      !Strain of Component (PTH)
!STRA_C_3      !Strain of Component (PTH)
!STRA_C_4      !Strain of Component (PTH)
!STRA_C_5      !Strain of Component (PTH)
!STRE_C_1      !Stress of Component (PTH)
!STRE_C_2      !Stress of Component (PTH)
!STRE_C_3      !Stress of Component (PTH)
!STRE_C_4      !Stress of Component (PTH)
!STRE_C_5      !Stress of Component (PTH)


!! board
! 30,95,110,125,150,270 (deg C)
!M_B_T1        !Temperature (deg K)
!M_B_T2        !Temperature (deg K)
!M_B_T3        !Temperature (deg K)
!M_B_T4        !Temperature (deg K)
!M_B_T5        !Temperature (deg K)
!M_B_T6        !Temperature (deg K)
!E_B_x1        !Young's Modulus of PWB (MPa)
!E_B_x2        !Young's Modulus of PWB (MPa)
!E_B_x3        !Young's Modulus of PWB (MPa)
!E_B_x4        !Young's Modulus of PWB (MPa)
!E_B_x5        !Young's Modulus of PWB (MPa)
!E_B_x6        !Young's Modulus of PWB (MPa)
!G_B_yz1       !Shear Modulus of PWB (MPa)
!G_B_yz2       !Shear Modulus of PWB (MPa)
!G_B_yz3       !Shear Modulus of PWB (MPa)
!G_B_yz4       !Shear Modulus of PWB (MPa)
!G_B_yz5       !Shear Modulus of PWB (MPa)
!G_B_yz6       !Shear Modulus of PWB (MPa)
!V_B_yz        !Poisson's Ratio of PWB
!CTE_B_x       !CTE of PWB (mm/mmK)
!E_B_y1        !Young's Modulus of PWB (MPa)
!E_B_y2        !Young's Modulus of PWB (MPa)
!E_B_y3        !Young's Modulus of PWB (MPa)
!E_B_y4        !Young's Modulus of PWB (MPa)
!E_B_y5        !Young's Modulus of PWB (MPa)
!E_B_y6        !Young's Modulus of PWB (MPa)
!G_B_zx1       !Shear Modulus of PWB (MPa)
!G_B_zx2       !Shear Modulus of PWB (MPa)
!G_B_zx3       !Shear Modulus of PWB (MPa)
!G_B_zx4       !Shear Modulus of PWB (MPa)
!G_B_zx5       !Shear Modulus of PWB (MPa)
!G_B_zx6       !Shear Modulus of PWB (MPa)
```

```
!V_B_zx         !Poisson's Ratio of PWB
!CTE_B_y1       !CTE of PWB (mm/mmK)
!CTE_B_y2       !CTE of PWB (mm/mmK)
!CTE_B_y3       !CTE of PWB (mm/mmK)
!CTE_B_y4       !CTE of PWB (mm/mmK)
!CTE_B_y5       !CTE of PWB (mm/mmK)
!CTE_B_y6       !CTE of PWB (mm/mmK)
!E_B_z1         !Young's Modulus of PWB (MPa)
!E_B_z2         !Young's Modulus of PWB (MPa)
!E_B_z3         !Young's Modulus of PWB (MPa)
!E_B_z4         !Young's Modulus of PWB (MPa)
!E_B_z5         !Young's Modulus of PWB (MPa)
!E_B_z6         !Young's Modulus of PWB (MPa)
!G_B_xy1        !Shear Modulus of PWB (MPa)
!G_B_xy2        !Shear Modulus of PWB (MPa)
!G_B_xy3        !Shear Modulus of PWB (MPa)
!G_B_xy4        !Shear Modulus of PWB (MPa)
!G_B_xy5        !Shear Modulus of PWB (MPa)
!G_B_xy6        !Shear Modulus of PWB (MPa)
!V_B_xy         !Poisson's Ratio of PWB
!CTE_B_z        !CTE of PWB (mm/mmK)

!! Geometric Parameters
D_H = 0.3429
TH_P = 0.0127
D_PAD = 0.508
TH_PAD = 0.04826
TH_B = 2.0

!! Operating Parameters
T_0 = 273+20.0
T_C0 = 273+20.0
T_C1 = 273+5.0
T_C2 = 273+90.0
T_B0 = 273+20.0
T_B1 = 273+5.0
T_B2 = 273+90.0
time_RMP = 1.0
time_DHI = 4.0
time_DLO = 18.0

!! Material Parameters

!! PTH(copper)
E_C = 120000.0
V_C = 0.3
G_C = 448.0
CTE_C = 1.7E-5
STRA_C_1 = 3.0E-4
STRA_C_2 = 0.0010
STRA_C_3 = 0.0040
STRA_C_4 = 0.01
STRA_C_5 = 0.02
STRE_C_1 = 30.0
STRE_C_2 = 110.0
STRE_C_3 = 186.0
STRE_C_4 = 217.0
STRE_C_5 = 234.0

!! board(FR4)
M_B_T1 = 303.0
M_B_T2 = 368.0
```

```
M_B_T3 = 383.0
M_B_T4 = 398.0
M_B_T5 = 423.0
M_B_T6 = 543.0
E_B_x1 = 22400.0
E_B_x2 = 20680.0
E_B_x3 = 19970.0
E_B_x4 = 19300.0
E_B_x5 = 17920.0
E_B_x6 = 16000.0
G_B_yz1 = 199.0
G_B_yz2 = 189.0
G_B_yz3 = 173.0
G_B_yz4 = 157.0
G_B_yz5 = 142.0
G_B_yz6 = 139.3
V_B_yz = 0.1425
CTE_B_x = 2.0E-5
E_B_y1 = 1600.0
E_B_y2 = 1200.0
E_B_y3 = 1100.0
E_B_y4 = 1000.0
E_B_y5 = 600.0
E_B_y6 = 450.0
G_B_zx1 = 630.0
G_B_zx2 = 600.0
G_B_zx3 = 550.0
G_B_zx4 = 500.0
G_B_zx5 = 450.0
G_B_zx6 = 441.0
V_B_zx = 0.136
CTE_B_y1 = 8.65E-5
CTE_B_y2 = 8.65E-5
CTE_B_y3 = 2.43E-4
CTE_B_y4 = 4.0E-4
CTE_B_y5 = 4.0E-4
CTE_B_y6 = 4.0E-4
E_B_z1 = 22400.0
E_B_z2 = 20680.0
E_B_z3 = 19970.0
E_B_z4 = 19300.0
E_B_z5 = 17920.0
E_B_z6 = 16000.0
G_B_xy1 = 199.0
G_B_xy2 = 189.0
G_B_xy3 = 173.0
G_B_xy4 = 157.0
G_B_xy5 = 142.0
G_B_xy6 = 139.3
V_B_xy = 0.1425
CTE_B_z = 2.0E-5

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.2 define element types
ET,1,PLANE42,,,1,,                    !For Plane82 element, keyopt(3) = 1 means axisymmetric

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.3 create Material models

!! pt_hole (copper)
UIMP,pt_hole,EX,EY,EZ,E_C,E_C,E_C
UIMP,pt_hole,NUXZ,NUXY,NUYZ,V_C,V_C,V_C
```

```
UIMP,pt_hole,ALPX,ALPY,ALPZ,CTE_C,CTE_C,CTE_C
UIMP,pt_hole,GXZ,GXY,GYZ,G_C,G_C,G_C
UIMP,pt_hole,REFT,,,T_0
UIMP,pt_hole,DENS,,,1 !dummy value
! MKIN table for copper Strain (mm/mm), stress (N/mm^2)
TB, MKIN, pt_hole,
TBTEMP, , STRAIN
TBDATA, 1, STRA_C_1, STRA_C_2, STRA_C_3, STRA_C_4, STRA_C_5
TBTEMP
TBDATA, 1, STRE_C_1, STRE_C_2, STRE_C_3, STRE_C_4, STRE_C_5

!! board (FR4)
MPTEMP         ! Clear material temperature table
MPTEMP,1,M_B_T1,M_B_T2,M_B_T3,M_B_T4,M_B_T5,M_B_T6
! Young's Modulus
MPDATA,EX,board,1,E_B_x1,E_B_x2,E_B_x3,E_B_x4,E_B_x5,E_B_x6
MPDATA,EY,board,1,E_B_y1,E_B_y2,E_B_y3,E_B_y4,E_B_y5,E_B_y6
MPDATA,EZ,board,1,E_B_z1,E_B_z2,E_B_z3,E_B_z4,E_B_z5,E_B_z6
! Poisson's Ratio
MPDATA,NUYZ,board,1,V_B_yz,V_B_yz,V_B_yz,V_B_yz,V_B_yz,V_B_yz
MPDATA,NUXZ,board,1,V_B_zx,V_B_zx,V_B_zx,V_B_zx,V_B_zx,V_B_zx
MPDATA,NUXY,board,1,V_B_xy,V_B_xy,V_B_xy,V_B_xy,V_B_xy,V_B_xy
! Coefficient of Thermal Expansion
MPDATA,ALPX,board,1,CTE_B_x,CTE_B_x,CTE_B_x,CTE_B_x,CTE_B_x,CTE_B_x
MPDATA,ALPY,board,1,CTE_B_y1,CTE_B_y2,CTE_B_y3,CTE_B_y4,CTE_B_y5,CTE_B_y6
MPDATA,ALPZ,board,1,CTE_B_z,CTE_B_z,CTE_B_z,CTE_B_z,CTE_B_z,CTE_B_z
! Shear Modulus
MPDATA,GYZ,board,1,G_B_yz1,G_B_yz2,G_B_yz3,G_B_yz4,G_B_yz5,G_B_yz6
MPDATA,GXZ,board,1,G_B_zx1,G_B_zx2,G_B_zx3,G_B_zx4,G_B_zx5,G_B_zx6
MPDATA,GXY,board,1,G_B_xy1,G_B_xy2,G_B_xy3,G_B_xy4,G_B_xy5,G_B_xy6
UIMP,board,DENS,,,1 !dummy value

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.4 build geometry

!!Left bottom corner is origin

!! Key points
a11_x = D_H/2.0 - TH_P
a11_y = (-1.0)* (TH_B/2.0) - TH_PAD
a21_x = D_H/2.0
a21_y = (-1.0)* (TH_B/2.0) - TH_PAD
a31_x = D_PAD/2.0
a31_y = (-1.0)* (TH_B/2.0) - TH_PAD
a12_x = D_H/2.0 - TH_P
a12_y = (-1.0)* (TH_B/2.0)
a22_x = D_H/2.0
a22_y = (-1.0)* (TH_B/2.0)
a32_x = D_PAD/2.0
a32_y = (-1.0)* (TH_B/2.0)
a42_x = D_PAD * (3.0/2.0)
a42_y = (-1.0)* (TH_B/2.0)
a13_x = D_H/2.0 - TH_P
a13_y = TH_B/2.0
a23_x = D_H/2.0
a23_y = TH_B/2.0
a33_x = D_PAD/2.0
a33_y = TH_B/2.0
a43_x = D_PAD * (3.0/2.0)
a43_y = TH_B/2.0
a14_x = D_H/2.0 - TH_P
a14_y = TH_B/2.0 + TH_PAD
```

```
a24_x = D_H/2.0
a24_y = TH_B/2.0 + TH_PAD
a34_x = D_PAD/2.0
a34_y = TH_B/2.0 + TH_PAD

!!Points
k,1,a11_x,a11_y,0
k,2,a21_x,a21_y,0
k,3,a31_x,a31_y,0
k,4,a12_x,a12_y,0
k,5,a22_x,a22_y,0
k,6,a32_x,a32_y,0
k,7,a42_x,a42_y,0
k,8,a13_x,a13_y,0
k,9,a23_x,a23_y,0
k,10,a33_x,a33_y,0
k,11,a43_x,a43_y,0
k,12,a14_x,a14_y,0
k,13,a24_x,a24_y,0
k,14,a34_x,a34_y,0

!!Lines
l,1,2          !line 1 (a11,a21)
l,2,3          !line 2 (a21,a31)
l,3,6          !line 3 (a31,a32)
l,6,5          !line 4 (a32,a22)
l,5,9          !line 5 (a22,a23)
l,9,10         !line 6 (a23,a33)
l,10,14        !line 7 (a33,a34)
l,14,13        !line 8 (a34,a24)
l,13,12        !line 9 (a24,a14)
l,12,8         !line 10 (a14,a13)
l,8,4          !line 11 (a13,a12)
l,4,1          !line 12 (a12,a11)
l,10,11        !line 13 (a33,a43)
l,11,7         !line 14 (a43,a42)
l,7,6          !line 15 (a42,a32)
l,2,5          !line 16 (a21,a22)
l,5,4          !line 17 (a22,a12)
l,8,9          !line 18 (a13,a23)
l,9,13         !line 19 (a23,a24)

!!Areas

!area right_plating_1 (a11,a21,a22,a12)
a,1,2,5,4

!area right_plating_2 (a21,a31,a32,a22)
a,2,3,6,5

!area right_plating_3 (a12,a22,a23,a13)
a,4,5,9,8

!area right_plating_4 (a13,a23,a24,a14)
a,8,9,13,12

!area right_plating_5 (a23,a33,a34,a24)
a,9,10,14,13

!area right_pwb_1 (a22,a32,a33,a23)
a,5,6,10,9
```

```
!area right_pwb_2 (a32,a42,a43,a33)
a,6,7,11,10

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.5 mesh

!! Divide Lines

!! x direction (right)

!! line 1
lsel,s,loc,x,a11_x-incc,a21_x+incc  ! line selection
lsel,r,loc,y,a11_y-incc,a11_y+incc  ! line selection
lesize,all,,,4*n

!! line 17
lsel,s,loc,x,a12_x-incc,a22_x+incc  ! line selection
lsel,r,loc,y,a12_y-incc,a12_y+incc  ! line selection
lesize,all,,,4*n

!! line 18
lsel,s,loc,x,a13_x-incc,a23_x+incc  ! line selection
lsel,r,loc,y,a13_y-incc,a13_y+incc  ! line selection
lesize,all,,,4*n

!! line 9
lsel,s,loc,x,a14_x-incc,a24_x+incc  ! line selection
lsel,r,loc,y,a14_y-incc,a14_y+incc  ! line selection
lesize,all,,,4*n

!! line 2
lsel,s,loc,x,a21_x-incc,a31_x+incc  ! line selection
lsel,r,loc,y,a21_y-incc,a21_y+incc  ! line selection
lesize,all,,,4*n

!! line 4
lsel,s,loc,x,a22_x-incc,a32_x+incc  ! line selection
lsel,r,loc,y,a22_y-incc,a22_y+incc  ! line selection
lesize,all,,,4*n

!! line 6
lsel,s,loc,x,a23_x-incc,a33_x+incc  ! line selection
lsel,r,loc,y,a23_y-incc,a23_y+incc  ! line selection
lesize,all,,,4*n

!! line 8
lsel,s,loc,x,a24_x-incc,a34_x+incc  ! line selection
lsel,r,loc,y,a24_y-incc,a24_y+incc  ! line selection
lesize,all,,,4*n

!! line 15
lsel,s,loc,x,a32_x-incc,a42_x+incc  ! line selection
lsel,r,loc,y,a32_y-incc,a32_y+incc  ! line selection
lesize,all,,,4*n

!! line 13
lsel,s,loc,x,a33_x-incc,a43_x+incc  ! line selection
lsel,r,loc,y,a33_y-incc,a33_y+incc  ! line selection
lesize,all,,,4*n

!! y direction (right)
```

```
!! line
lsel,s,loc,x,a11_x-incc,a11_x+incc  ! line selection
lsel,r,loc,y,a11_y-incc,a12_y+incc  ! line selection
lesize,all,,,4*n

!! line
lsel,s,loc,x,a21_x-incc,a21_x+incc  ! line selection
lsel,r,loc,y,a21_y-incc,a22_y+incc  ! line selection
lesize,all,,,4*n

!! line
lsel,s,loc,x,a31_x-incc,a31_x+incc  ! line selection
lsel,r,loc,y,a31_y-incc,a32_y+incc  ! line selection
lesize,all,,,4*n

!! line
lsel,s,loc,x,a12_x-incc,a12_x+incc  ! line selection
lsel,r,loc,y,a12_y-incc,a13_y+incc  ! line selection
lesize,all,,,40*n

!! line
lsel,s,loc,x,a22_x-incc,a22_x+incc  ! line selection
lsel,r,loc,y,a22_y-incc,a23_y+incc  ! line selection
lesize,all,,,40*n

!! line
lsel,s,loc,x,a32_x-incc,a32_x+incc  ! line selection
lsel,r,loc,y,a32_y-incc,a33_y+incc  ! line selection
lesize,all,,,40*n

!! line
lsel,s,loc,x,a42_x-incc,a42_x+incc  ! line selection
lsel,r,loc,y,a42_y-incc,a43_y+incc  ! line selection
lesize,all,,,40*n

!! line
lsel,s,loc,x,a13_x-incc,a13_x+incc  ! line selection
lsel,r,loc,y,a13_y-incc,a14_y+incc  ! line selection
lesize,all,,,4*n

!! line
lsel,s,loc,x,a23_x-incc,a23_x+incc  ! line selection
lsel,r,loc,y,a23_y-incc,a24_y+incc  ! line selection
lesize,all,,,4*n

!! line
lsel,s,loc,x,a33_x-incc,a33_x+incc  ! line selection
lsel,r,loc,y,a33_y-incc,a34_y+incc  ! line selection
lesize,all,,,4*n

allsel
lplot

!! Assign the materials to the areas

asel,s,area,,1
aatt,1, ,1,0

asel,s,area,,2
aatt,1, ,1,0

asel,s,area,,3
```

```
aatt,1, ,1,0

asel,s,area,,4
aatt,1, ,1,0

asel,s,area,,5
aatt,1, ,1,0

asel,s,area,,6
aatt,2, ,1,0

asel,s,area,,7
aatt,2, ,1,0

allsel
aglue,all
aplot

!! Mesh
mshkey,1                ! mapped mesh
amesh,all               ! mesh

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.6 Boundary Conditions and Loads


!!Select appropriate nodes and apply BC

nsel,s,loc,x,a42_x-incc,a42_x+incc ! line selection
nsel,r,loc,y,0.0-incc,0.0+incc
d,all,uy,0

nummrg,all,1e-3
numcmp,all

allsel,all
finish

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 3. solve

!!  Solve

/solu

t_room          = T_C0
t_low           = T_C1
t_high          = T_C2

time_VAR = 0
time_STEP = 3600

antype,trans,new
nropt,auto,,                    !Specifies the Newton-Raphson options in a static or full transient analysis

! load step 0
tref,t_room
time_VAR = time_VAR + time_STEP*time_RMP
time, time_VAR
kbc,0                           !specifies stepped or ramped loading within a load step
nsubst,20                       !specifies the number of sub steps to be taken this load step.
```

```
autots,off                  !specifies whether to use automatic time stepping or load stepping
nropt,full,,on              !use full Newton-Raphson with adaptive descent
sstif,on                    !include stress stiffening
nlgeom,on                   !include large deformation effects
eqslv,sparse                !specifies the type of equation solver
toffst,0,                   !set the temp offset from absolute zero to zero
neqit,100                   !set 100 as max number of iterations
allsel
bf,all,temp,t_low           !temperature
outres,all,last             !put only end of time step results to the database
solve
save

! load step 1-1
time_VAR = time_VAR + time_STEP*time_DLO
time, time_VAR
kbc,0                       !specifies stepped or ramped loading within a load step
nsubst,20                   !specifies the number of sub steps to be taken this load step.
autots,off                  !specifies whether to use automatic time stepping or load stepping
nropt,full,,on              !use full Newton-Raphson with adaptive descent
sstif,on                    !include stress stiffening
nlgeom,on                   !include large deformation effects
eqslv,sparse                !specifies the type of equation solver
toffst,0,                   !set the temp offset from absolute zero to zero
neqit,100                   !set 100 as max number of iterations
allsel
bf,all,temp,t_low           !temperature
outres,all,last             !put only end of time step results to the database
solve
save

! load step 1-2
time_VAR = time_VAR + time_STEP*time_RMP
time, time_VAR
kbc,0                       !specifies stepped or ramped loading within a load step
nsubst,20                   !specifies the number of sub steps to be taken this load step.
autots,off                  !specifies whether to use automatic time stepping or load stepping
nropt,full,,on              !use full Newton-Raphson with adaptive descent
sstif,on                    !include stress stiffening
nlgeom,on                   !include large deformation effects
eqslv,sparse                !specifies the type of equation solver
toffst,0,                   !set the temp offset from absolute zero to zero
neqit,100                   !set 100 as max number of iterations
allsel
bf,all,temp,t_high          !temperature
outres,all,last             !put only end of time step results to the database
solve
save

! load step 1-3
time_VAR = time_VAR + time_STEP*time_DHI
time, time_VAR
kbc,0                       !specifies stepped or ramped loading within a load step
nsubst,20                   !specifies the number of sub steps to be taken this load step.
autots,off                  !specifies whether to use automatic time stepping or load stepping
nropt,full,,on              !use full Newton-Raphson with adaptive descent
sstif,on                    !include stress stiffening
nlgeom,on                   !include large deformation effects
eqslv,sparse                !specifies the type of equation solver
toffst,0,                   !set the temp offset from absolute zero to zero
neqit,100                   !set 100 as max number of iterations
allsel
```

```
bf,all,temp,t_high                  !temperature
outres,all,last                     !put only end of time step results to the database
solve
save

finish

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 4. postprocess

!! Postprocessing

/post1

set,first
nsel,s,loc,x,a11_x-incc, a21_x+incc
nsel,r,loc,y,-0.04, +0.04
esln
esel,r,mat,,1
etable,pstrn1,epto,y
etable,area1,volu
sabs,1
smult,plst1,pstrn1,area1
pretab,pstrn1,area1,plst1
allsel

set,last
nsel,s,loc,x,a11_x-incc, a21_x+incc
nsel,r,loc,y,-0.04, +0.04
esln
esel,r,mat,,1
etable,pstrn2,epto,y
etable,area2,volu
sabs,1
smult,plst2,pstrn2,area2
pretab,pstrn2,area2,plst2

sadd,area,area1,area2,1/2,1/2

*if,T_C0,le,T_C1,AND,T_C0,le,T_C2,then
sadd,delplst,plst2,plst1,1,-1
*elseif,T_C0,gt,T_C1,AND,T_C0,le,T_C2,then
sadd,delplst,plst2,plst1,1,1
*else
sadd,delplst,plst2,plst1,1,-1
*endif

ssum
*get,sumplst,ssum,,item,delplst
*get,sumarea,ssum,,item,area
plstavg=sumplst/sumarea

*CFOPEN,'pred_pth_tm_s1','txt',' '
*VWRITE,plstavg, , , , , , , , ,
(F10.8)

!! Postprocessing
allsel
esel,r,mat,,1
PLNSOL,epto,y,0,1
```

C.3 A Thermo-Mechanical Fatigue Analysis Model for Solder Joints

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!The Two Dimensional Numerical Fatigue Analysis Model
!!of Solder Joint (Plane Strain Model)

!!by Injoong Kim
!!10/13/05

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 1. startup

finish
/clear,nostart
/title, Solder Joint Reliability

/RGB,INDEX,100,100,100, 0
/RGB,INDEX, 80, 80, 80,13
/RGB,INDEX, 60, 60, 60,14
/RGB,INDEX, 0, 0, 0,15
/REPLOT


!Units are mm,N/mm^2 (same as MPa) etc.
!Units

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2. preprocessor

/prep7

!! preference

KEYW,PR_SET,1
KEYW,PR_STRUC,1
KEYW,PR_THERM,1
KEYW,PR_MULTI,1

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.1 define parameters

!! Constants
incc = 1e-6       !tiny increment for selecting objects
n = 8             !mesh conversion factor 1/2/3/4

component=1
board=2
solder_joint=3

!! Geometric Parameters
!H_SJ             !Solder Joint Standoff Height (mm)
!H_F              !Fillet Height of Solder Joint (mm)
!L_S              !Base Length of Solder Joint (mm)
!L_C              !Length of Component (mm)
!H_C              !Height of Component (mm)
!TH_B             !Thickness of PWB (mm)
```

```
!! Operating Parameters
!T_0            !Ambient Temperature (deg K)
!T_C0           !Steady State Temperature of Component (deg K)
!T_C1           !State 1 Temperature of Component (deg K)
!T_C2           !State 2 Temperature of Component (deg K)
!T_B0           !Steady State Temperature of of PWB (deg K)
!T_B1           !State 1 Temperature of PWB (deg K)
!T_B2           !State 2 Temperature of PWB (deg K)

!time_RMP       !Ramp hour (Hour)
!time_DHI       !Duration hour at high temperature  (Hour)
!time_DLO       !Duration hour at low temperature   (Hour)

!! Material Parameters
!! component

!component=1

!E_C            !Young's Modulus of Component (MPa)
!V_C            !Poisson's Ratio of Component
!CTE_C          !CTE of Component (mm/mmK)

!! board

! 30,95,110,125,150,270 (deg C)
!M_B_T1         !Temperature (deg K)
!M_B_T2         !Temperature (deg K)
!M_B_T3         !Temperature (deg K)
!M_B_T4         !Temperature (deg K)
!M_B_T5         !Temperature (deg K)
!M_B_T6         !Temperature (deg K)

!E_B_x1         !Young's Modulus of PWB (MPa)
!E_B_x2         !Young's Modulus of PWB (MPa)
!E_B_x3         !Young's Modulus of PWB (MPa)
!E_B_x4         !Young's Modulus of PWB (MPa)
!E_B_x5         !Young's Modulus of PWB (MPa)
!E_B_x6         !Young's Modulus of PWB (MPa)

!G_B_yz1        !Shear Modulus of PWB (MPa)
!G_B_yz2        !Shear Modulus of PWB (MPa)
!G_B_yz3        !Shear Modulus of PWB (MPa)
!G_B_yz4        !Shear Modulus of PWB (MPa)
!G_B_yz5        !Shear Modulus of PWB (MPa)
!G_B_yz6        !Shear Modulus of PWB (MPa)

!V_B_yz         !Poisson's Ratio of PWB

!CTE_B_x        !CTE of PWB (mm/mmK)

!E_B_y1         !Young's Modulus of PWB (MPa)
!E_B_y2         !Young's Modulus of PWB (MPa)
!E_B_y3         !Young's Modulus of PWB (MPa)
!E_B_y4         !Young's Modulus of PWB (MPa)
!E_B_y5         !Young's Modulus of PWB (MPa)
!E_B_y6         !Young's Modulus of PWB (MPa)

!G_B_zx1        !Shear Modulus of PWB (MPa)
!G_B_zx2        !Shear Modulus of PWB (MPa)
!G_B_zx3        !Shear Modulus of PWB (MPa)
!G_B_zx4        !Shear Modulus of PWB (MPa)
```

```
!G_B_zx5        !Shear Modulus of PWB (MPa)
!G_B_zx6        !Shear Modulus of PWB (MPa)

!V_B_zx         !Poisson's Ratio of PWB

!CTE_B_y1       !CTE of PWB (mm/mmK)
!CTE_B_y2       !CTE of PWB (mm/mmK)
!CTE_B_y3       !CTE of PWB (mm/mmK)
!CTE_B_y4       !CTE of PWB (mm/mmK)
!CTE_B_y5       !CTE of PWB (mm/mmK)
!CTE_B_y6       !CTE of PWB (mm/mmK)

!E_B_z1         !Young's Modulus of PWB (MPa)
!E_B_z2         !Young's Modulus of PWB (MPa)
!E_B_z3         !Young's Modulus of PWB (MPa)
!E_B_z4         !Young's Modulus of PWB (MPa)
!E_B_z5         !Young's Modulus of PWB (MPa)
!E_B_z6         !Young's Modulus of PWB (MPa)

!G_B_xy1        !Shear Modulus of PWB (MPa)
!G_B_xy2        !Shear Modulus of PWB (MPa)
!G_B_xy3        !Shear Modulus of PWB (MPa)
!G_B_xy4        !Shear Modulus of PWB (MPa)
!G_B_xy5        !Shear Modulus of PWB (MPa)
!G_B_xy6        !Shear Modulus of PWB (MPa)

!V_B_xy         !Poisson's Ratio of PWB

!CTE_B_z        !CTE of PWB (mm/mmK)

!! solder joint

! -25,25,60,100,150,227 (deg C)
!M_SJ_T1        !Temperature (deg K)
!M_SJ_T2        !Temperature (deg K)
!M_SJ_T3        !Temperature (deg K)
!M_SJ_T4        !Temperature (deg K)
!M_SJ_T5        !Temperature (deg K)
!M_SJ_T6        !Temperature (deg K)

!E_SJ_x1        !Young's Modulus of Solder (MPa)
!E_SJ_x2        !Young's Modulus of Solder (MPa)
!E_SJ_x3        !Young's Modulus of Solder (MPa)
!E_SJ_x4        !Young's Modulus of Solder (MPa)
!E_SJ_x5        !Young's Modulus of Solder (MPa)
!E_SJ_x6        !Young's Modulus of Solder (MPa)

!V_SJ           !Poisson's Ratio of Solder

!CTE_SJ         !CTE of Solder (mm/mmK)

! (1)So  (2)Q/R  (3)A   (4)Xi  (5)m   (6)Ho  (7)Sh  (8)n  (9)a

!So_SJ                  !initial value of deformation resistance (MPa)
!QR_SJ                  !Q: activation energy and R: universal gas constant (1/K)
!A_Const_SJ             !pre-exponential factor (1/sec)
!Xi_SJ                  !multiplier of stress
!m_SJ                   !strain rate sensitivity of stress
!Ho_SJ                  !hardening / softening constant (MPa)
!Sh_SJ                  !coefficient for deformation resistance saturation value (MPa)
!n_SJ                   !strain rate sensitivity of saturation (deformation resistance) value
!a_SJ                   !strain rate sensitivity of hardening or softening
```

!! Geometric Parameters
H_SJ = 0.127
H_F = 0.4
L_S = 0.3
L_C = 2.0
H_C = 0.5
TH_B = 2.0

!! Operating Parameters
T_0 = 273+20.0
T_C0 = 273+20.0
T_C1 = 273+5.0
T_C2 = 273+90.0
T_B0 = 273+20.0
T_B1 = 273+5.0
T_B2 = 273+90.0
time_RMP = 1.0
time_DHI = 4.0
time_DLO = 18.0

!! Material Parameters

!! Component
E_C = 255110.0
V_C = 0.3
G_C = 98119.0
CTE_C = 6.7E-6

!! board(FR4)
M_B_T1 = 303.0
M_B_T2 = 368.0
M_B_T3 = 383.0
M_B_T4 = 398.0
M_B_T5 = 423.0
M_B_T6 = 543.0
E_B_x1 = 22400.0
E_B_x2 = 20680.0
E_B_x3 = 19970.0
E_B_x4 = 19300.0
E_B_x5 = 17920.0
E_B_x6 = 16000.0
G_B_yz1 = 199.0
G_B_yz2 = 189.0
G_B_yz3 = 173.0
G_B_yz4 = 157.0
G_B_yz5 = 142.0
G_B_yz6 = 139.3
V_B_yz = 0.1425
CTE_B_x = 2.0E-5
E_B_y1 = 1600.0
E_B_y2 = 1200.0
E_B_y3 = 1100.0
E_B_y4 = 1000.0
E_B_y5 = 600.0
E_B_y6 = 450.0
G_B_zx1 = 630.0
G_B_zx2 = 600.0
G_B_zx3 = 550.0
G_B_zx4 = 500.0
G_B_zx5 = 450.0
G_B_zx6 = 441.0

```
V_B_zx = 0.136
CTE_B_y1 = 8.65E-5
CTE_B_y2 = 8.65E-5
CTE_B_y3 = 2.43E-4
CTE_B_y4 = 4.0E-4
CTE_B_y5 = 4.0E-4
CTE_B_y6 = 4.0E-4
E_B_z1 = 22400.0
E_B_z2 = 20680.0
E_B_z3 = 19970.0
E_B_z4 = 19300.0
E_B_z5 = 17920.0
E_B_z6 = 16000.0
G_B_xy1 = 199.0
G_B_xy2 = 189.0
G_B_xy3 = 173.0
G_B_xy4 = 157.0
G_B_xy5 = 142.0
G_B_xy6 = 139.3
V_B_xy = 0.1425
CTE_B_z = 2.0E-5

!! solder joint
M_SJ_T1 = 248.0
M_SJ_T2 = 298.0
M_SJ_T3 = 333.0
M_SJ_T4 = 373.0
M_SJ_T5 = 423.0
M_SJ_T6 = 500.0
E_SJ_x1 = 58881.0
E_SJ_x2 = 49229.0
E_SJ_x3 = 42472.0
E_SJ_x4 = 34750.0
E_SJ_x5 = 25097.0
E_SJ_x6 = 10232.0
V_SJ = 0.4
CTE_SJ = 2.4E-5
So_SJ = 39.9
QR_SJ = 8900.0
A_Const_SJ = 22300.0
Xi_SJ = 6.0
m_SJ = 0.182
Ho_SJ = 3321.15
Sh_SJ = 73.81
n_SJ = 0.018
a_SJ = 1.8199999999999998

!!!
```

```
UIMP,component,DENS,,,1 !dummy value

!! Board
MPTEMP        ! Clear material temperature table
MPTEMP,1,M_B_T1,M_B_T2,M_B_T3,M_B_T4,M_B_T5,M_B_T6

! Young's Modulus
MPDATA,EX,board,1,E_B_x1,E_B_x2,E_B_x3,E_B_x4,E_B_x5,E_B_x6
MPDATA,EY,board,1,E_B_y1,E_B_y2,E_B_y3,E_B_y4,E_B_y5,E_B_y6
MPDATA,EZ,board,1,E_B_z1,E_B_z2,E_B_z3,E_B_z4,E_B_z5,E_B_z6

! Poisson's Ratio
MPDATA,NUYZ,board,1,V_B_yz,V_B_yz,V_B_yz,V_B_yz,V_B_yz,V_B_yz
MPDATA,NUXZ,board,1,V_B_zx,V_B_zx,V_B_zx,V_B_zx,V_B_zx,V_B_zx
MPDATA,NUXY,board,1,V_B_xy,V_B_xy,V_B_xy,V_B_xy,V_B_xy,V_B_xy

! Coefficient of Thermal Expansion
MPDATA,ALPX,board,1,CTE_B_x,CTE_B_x,CTE_B_x,CTE_B_x,CTE_B_x,CTE_B_x
MPDATA,ALPY,board,1,CTE_B_y1,CTE_B_y2,CTE_B_y3,CTE_B_y4,CTE_B_y5,CTE_B_y6
MPDATA,ALPZ,board,1,CTE_B_z,CTE_B_z,CTE_B_z,CTE_B_z,CTE_B_z,CTE_B_z

! SHEAR MODULUS
MPDATA,GYZ,board,1,G_B_yz1,G_B_yz2,G_B_yz3,G_B_yz4,G_B_yz5,G_B_yz6
MPDATA,GXZ,board,1,G_B_zx1,G_B_zx2,G_B_zx3,G_B_zx4,G_B_zx5,G_B_zx6
MPDATA,GXY,board,1,G_B_xy1,G_B_xy2,G_B_xy3,G_B_xy4,G_B_xy5,G_B_xy6
UIMP,board,DENS,,,1 !dummy value

!! Solder Joint
!Sn(96.5)-Ag(3.5) Solder (Viscoplastic model using Anands)
MPTEMP                              ! Clear material temperature table
MPTEMP,1,M_SJ_T1,M_SJ_T2,M_SJ_T3,M_SJ_T4,M_SJ_T5,M_SJ_T6
! Young's modulus (N/mm^2)
MPDATA,EX,solder_joint,1,E_SJ_x1,E_SJ_x2,E_SJ_x3,E_SJ_x4,E_SJ_x5,E_SJ_x6
MP,ALPX,solder_joint,CTE_SJ         ! Coefficient of thermal expansion (mm/mm C)
MP,NUXY,solder_joint,V_SJ           ! Poisson's ratio
TB,ANAND, solder_joint
! (1)So  (2)Q/R  (3)A   (4)Xi  (5)m   (6)Ho  (7)Sh  (8)n  (9)a
TBDATA,1, So_SJ, QR_SJ, A_Const_SJ, Xi_SJ, m_SJ, Ho_SJ
TBDATA,7, Sh_SJ, n_SJ, a_SJ
UIMP,solder_joint,DENS,,,1 !dummy value



!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.4 build geometry

!!Left bottom corner is origin

!! Key points
p11_x = 0.0
p11_y = 0.0
p12_x =          0.0
p12_y = TH_B
p13_x = 0.0
p13_y = TH_B + H_SJ
p14_x = 0.0
p14_y = TH_B + H_SJ + H_C
p22_x = L_C/2.0 - L_S/2.0
p22_y = TH_B
p23_x = L_C/2.0 - L_S/2.0
p23_y = TH_B + H_SJ
p33_x = L_C/2.0
```

```
p33_y = TH_B + H_SJ
p33_a_x = L_C/2.0
p33_a_y = TH_B + H_SJ + H_F
p34_x = L_C/2.0
p34_y = TH_B + H_SJ + H_C
p42_x = L_C/2.0 + L_S/2.0
p42_y = TH_B
p43_x = L_C/2.0 + L_S/2.0
p43_y = TH_B + H_SJ
p51_x = L_C/2.0 + 2.0*L_S
p51_y = 0.0
p52_x = L_C/2.0 + 2.0*L_S
p52_y = TH_B

!!Points
k,1,p11_x,p11_y,0
k,2,p12_x,p12_y,0
k,3,p13_x,p13_y,0
k,4,p14_x,p14_y,0
k,5,p22_x,p22_y,0
k,6,p23_x,p23_y,0
k,7,p33_x,p33_y,0
k,8,p33_a_x,p33_a_y,0
k,9,p34_x,p34_y,0
k,10,p42_x,p42_y,0
k,11,p43_x,p43_y,0
k,12,p51_x,p51_y,0
k,13,p52_x,p52_y,0

!!Lines
l,1,2                    !line 1 (11,12)
l,2,5                    !line 2 (12,22)
l,5,10                   !line 3 (22,42)
l,10,13                  !line 4 (42,52)
l,13,12                  !line 5 (52,51)
l,12,1                   !line 6 (51,11)
l,10,11                  !line 7 (42,43)
l,11,8                   !line 8 (43,33_a)
l,8,7                    !line 9 (33_a,33)
l,7,6                    !line 10 (33,23)
l,6,5                    !line 11 (23,22)
l,6,3                    !line 12 (23,13)
l,3,4                    !line 13 (13,14)
l,4,9                    !line 14 (14,34)
l,9,8                    !line 15 (34,33_a)

!!Areas
a,3,4,9,8,7,6            !area 1:component (13,14,34,33_a,33,23)
a,1,2,5,10,13,12         !area 2:board (11,12,22,42,52,51)
a,5,6,7,8,11,10          !area 3:solder joint(22,23,33,33_a,43,42)


!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.5 mesh


!! Divide Lines

!! line 1
lsel,s,loc,x,p11_x-incc,p11_x+incc  ! line selection
lsel,r,loc,y,p11_y-incc,p12_y+incc  ! line selection
lesize,all,,,4*n
```

```
!! line 5
lsel,s,loc,x,p51_x-incc,p51_x+incc  ! line selection
lsel,r,loc,y,p51_y-incc,p52_y+incc  ! line selection
lesize,all,,,4*n

!! line 11
lsel,s,loc,x,p22_x-incc,p22_x+incc  ! line selection
lsel,r,loc,y,p22_y-incc,p23_y+incc  ! line selection
lesize,all,,,3

!! line 7
lsel,s,loc,x,p42_x-incc,p42_x+incc  ! line selection
lsel,r,loc,y,p42_y-incc,p43_y+incc  ! line selection
lesize,all,,,3

!! line 13
lsel,s,loc,x,p13_x-incc,p13_x+incc  ! line selection
lsel,r,loc,y,p13_y-incc,p14_y+incc  ! line selection
lesize,all,,,2*n

!! line 9
lsel,s,loc,x,p33_x-incc,p33_x+incc  ! line selection
lsel,r,loc,y,p33_y-incc,p33_a_y+incc            ! line selection
lesize,all,,,4*n

!! line 15
lsel,s,loc,x,p33_a_x-incc,p33_a_x+incc        ! line selection
lsel,r,loc,y,p33_a_y-incc,p34_y+incc            ! line selection
lesize,all,,,2*n

!! line 8
lsel,s,loc,x,p33_a_x-incc,p43_x+incc            ! line selection
lsel,r,loc,y,p43_y-incc,p33_a_y+incc            ! line selection
lesize,all,,,4*n

!! line 6
lsel,s,loc,x,p11_x-incc,p51_x+incc  ! line selection
lsel,r,loc,y,p11_y-incc,p11_y+incc  ! line selection
lesize,all,,,4*n

!! line 2
lsel,s,loc,x,p12_x-incc,p22_x+incc  ! line selection
lsel,r,loc,y,p12_y-incc,p12_y+incc  ! line selection
lesize,all,,,2*n

!! line 2
lsel,s,loc,x,p22_x-incc,p42_x+incc  ! line selection
lsel,r,loc,y,p22_y-incc,p22_y+incc  ! line selection
lesize,all,,,4*n

!! line 4
lsel,s,loc,x,p42_x-incc,p52_x+incc  ! line selection
lsel,r,loc,y,p42_y-incc,p42_y+incc  ! line selection
lesize,all,,,2*n

!! line 12
lsel,s,loc,x,p13_x-incc,p23_x+incc  ! line selection
lsel,r,loc,y,p13_y-incc,p13_y+incc  ! line selection
lesize,all,,,2*n

!! line 10
```

```
lsel,s,loc,x,p23_x-incc,p33_x+incc  ! line selection
lsel,r,loc,y,p23_y-incc,p23_y+incc  ! line selection
lesize,all,,,2*n

!! line 14
lsel,s,loc,x,p14_x-incc,p34_x+incc  ! line selection
lsel,r,loc,y,p14_y-incc,p14_y+incc  ! line selection
lesize,all,,,2*n

allsel
lplot

!! Assign the materials to the areas

asel,s,area,,1
aatt,1, ,1,0

asel,s,area,,2
aatt,2, ,1,0

asel,s,area,,3
aatt,3, ,2,0

!! Glue
allsel
aglue,all
aplot

!! Mesh

amesh,all                    ! mesh

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.6 Boundary Conditions and Loads


!!Select apropriate nodes and apply BC

nsel,s,loc,x,p11_x-incc,p11_x+incc ! line selection
d,all,ux,0

nsel,s,loc,x,p11_x-incc,p11_x+incc ! node selection
nsel,r,loc,y,p11_y-incc,p11_y+incc
d,all,uy,0

nummrg,all,1e-3
numcmp,all


nlgeom, on

allsel,all
finish


!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 3. solve

!!  Solve

/solu
```

```
t_room = T_C0
t_low          = T_C1
t_high         = T_C2

time_VAR = 0
time_STEP = 3600

antype,trans,new
nropt,auto,,                    !Specifies the Newton-Raphson options in a static or full transient analysis

! load step 0
tref,t_room
time_VAR = time_VAR + time_STEP*time_RMP
time, time_VAR
kbc,0                           !specifies stepped or ramped loading within a load step
nsubst,20                       !specifies the number of substeps to be taken this load step.
autots,off                      !specifies whether to use automatic time stepping or load stepping
nropt,full,,on                  !use full Newton-Raphson with adaptive descent
sstif,on                        !include stress stiffening
nlgeom,on                       !include large deformation effects
eqslv,sparse                    !specifies the type of equation solver
toffst,0,                       !set the temp offset from absolute zero to zero
neqit,100                        !set 100 as max number of iterations
allsel
bf,all,temp,t_low               !temperature
outres,all,last                 !put only end of time step results to the database
solve
save

! load step 1-1
time_VAR = time_VAR + time_STEP*time_DLO
time, time_VAR
kbc,0                           !specifies stepped or ramped loading within a load step
nsubst,20                       !specifies the number of substeps to be taken this load step.
autots,off                      !specifies whether to use automatic time stepping or load stepping
nropt,full,,on                  !use full Newton-Raphson with adaptive descent
sstif,on                        !include stress stiffening
nlgeom,on                       !include large deformation effects
eqslv,sparse                    !specifies the type of equation solver
toffst,0,                       !set the temp offset from absolute zero to zero
neqit,100                       !set 100 as max number of iterations
allsel
bf,all,temp,t_low               !temperature
outres,all,last                 !put only end of time step results to the database
solve
save

! load step 1-2
time_VAR = time_VAR + time_STEP*time_RMP
time, time_VAR
kbc,0                           !specifies stepped or ramped loading within a load step
nsubst,20                       !specifies the number of substeps to be taken this load step.
autots,off                      !specifies whether to use automatic time stepping or load stepping
nropt,full,,on                  !use full Newton-Raphson with adaptive descent
sstif,on                        !include stress stiffening
nlgeom,on                       !include large deformation effects
eqslv,sparse                    !specifies the type of equation solver
toffst,0,                       !set the temp offset from absolute zero to zero
neqit,100                       !set 100 as max number of iterations
allsel
bf,all,temp,t_high              !temperature
```

```
outres,all,last                 !put only end of time step results to the database
solve
save

! load step 1-3
time_VAR = time_VAR + time_STEP*time_DHI
time, time_VAR
kbc,0                           !specifies stepped or ramped loading within a load step
nsubst,20                       !specifies the number of substeps to be taken this load step.
autots,off                      !specifies whether to use automatic time stepping or load stepping
nropt,full,,on                  !use full Newton-Raphson with adaptive descent
sstif,on                        !include stress stiffening
nlgeom,on                       !include large deformation effects
eqslv,sparse                    !specifies the type of equation solver
toffst,0,                       !set the temp offset from absolute zero to zero
neqit,100                       !set 100 as max number of iterations
allsel
bf,all,temp,t_high              !temperature
outres,all,last                 !put only end of time step results to the database
solve
save


finish
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 4. postprocess

!! Postprocessing
/post1

set,2
nsel,s,loc,x,p33_x-0.04, p33_x+0.04
nsel,r,loc,y,p33_y-0.04, p33_y+0.04
esln
esel,r,mat,,3
etable,tstrn1,epto,xy
etable,area1,volu
sabs,1
smult,plst1,tstrn1,area1
pretab,tstrn1,area1,plst1
allsel

set,last
nsel,s,loc,x,p33_x-0.04, p33_x+0.04
nsel,r,loc,y,p33_y-0.04, p33_y+0.04
esln
esel,r,mat,,3
etable,tstrn2,epto,xy
etable,area2,volu
sabs,1
smult,plst2,tstrn2,area2
pretab,tstrn2,area2,plst2

sadd,area,area1,area2,1/2,1/2

*if,T_C0,le,T_C1,AND,T_C0,le,T_C2,then
sadd,delplst,plst2,plst1,1,-1
*elseif,T_C0,gt,T_C1,AND,T_C0,le,T_C2,then
sadd,delplst,plst2,plst1,1,1
*else
sadd,delplst,plst2,plst1,1,-1
```

```
*endif

ssum
*get,sumplst,ssum,,item,delplst
*get,sumarea,ssum,,item,area
plstavg=sumplst/sumarea
*CFOPEN,'pred_sj_tm_s1','txt',' '
*VWRITE,plstavg, , , , , , , ,
(F10.8)

!! Postprocessing
allsel
esel,r,mat,,3
PLNSOL,epto,XY,0,1
```

C.4 A Thermo-Mechanical Fatigue Analysis Model for Solder Balls

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!The Three Dimensional Numerical Fatigue Analysis Model
!!of BGA (Generalized Plane Deformation Model)

!!by Injoong Kim
!!12/13/05

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 1. startup

finish
/clear,nostart
/title, CBGA Reliability
/RGB,INDEX,100,100,100, 0
/RGB,INDEX, 80, 80, 80,13
/RGB,INDEX, 60, 60, 60,14
/RGB,INDEX, 0, 0, 0,15
/REPLOT

!Units are mm,N/mm^2 (same as MPa) etc.
!Units

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2. preprocessor

/prep7
!! preference

KEYW,PR_SET,1
KEYW,PR_STRUC,1
KEYW,PR_THERM,1
KEYW,PR_MULTI,1


!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.1 define parameters

!! Constants
incc = 1e-6                !tiny increment for selecting objects
m_num = 4
m_num_sb = 2

CBGA=1
board=2
solder_ball=3
pad=4

!! Geometric Parameters
!L_CBGA                    !Size of CBGA (mm)
!Eff_L_CBGA                !Effective Size of CBGA (mm)
!Th_CBGA                   !Thickness of CBGA (mm)
!Th_PWB                    !Thickness of PWB (mm)
!P_sb                      !Pitch of Solder Balls (mm)
!Eff_P_sb                  !Effective Pitch of Solder Balls (mm)
!Pos_First_sb              !Position of First Solder Balls (mm)
```

```
!N_sb                   !Number of Solder Balls (mm)
!Eff_N_sb               !Effective Number of Solder Balls (mm)
!D_sb                   !Solder Balls Diameter(mm)
!H_sb                   !Solder Balls Height(mm)
!D_b_pad                !Board Pad Diameter(mm)
!D_c_pad                !CBGA Pad Diameter(mm)
!Th_pad                  !Pad Thickness(mm)

!! Operating Parameters
!T_0                    !Ambient Temperature (deg K)
!T_C0                   !Steady State Temperature of Component (deg K)
!T_C1                   !State 1 Temperature of Component (deg K)
!T_C2                   !State 2 Temperature of Component (deg K)
!T_B0                   !Steady State Temperature of of PWB (deg K)
!T_B1                   !State 1 Temperature of PWB (deg K)
!T_B2                   !State 2 Temperature of PWB (deg K)
!time_RMP               !Ramp hour (Hour)
!time_DHI               !Duration hour at high temperature  (Hour)
!time_DLO               !Duration hour at low temperature   (Hour)

!! Material Parameters
!! CBGA
!E_C                    !Young's Modulus of CBGA (MPa)
!V_C                    !Poisson's Ratio of CBGA
!CTE_C                  !CTE of CBGA (mm/mmK)

!! board
!M_B_T1                 !Temperature (deg K)
!M_B_T2                 !Temperature (deg K)
!M_B_T3                 !Temperature (deg K)
!M_B_T4                 !Temperature (deg K)
!M_B_T5                 !Temperature (deg K)
!M_B_T6                 !Temperature (deg K)
!! board x direction
!E_B_x1                 !Young's Modulus of PWB (MPa)
!E_B_x2                 !Young's Modulus of PWB (MPa)
!E_B_x3                 !Young's Modulus of PWB (MPa)
!E_B_x4                 !Young's Modulus of PWB (MPa)
!E_B_x5                 !Young's Modulus of PWB (MPa)
!E_B_x6                 !Young's Modulus of PWB (MPa)
!G_B_yz1                !Shear Modulus of PWB (MPa)
!G_B_yz2                !Shear Modulus of PWB (MPa)
!G_B_yz3                !Shear Modulus of PWB (MPa)
!G_B_yz4                !Shear Modulus of PWB (MPa)
!G_B_yz5                !Shear Modulus of PWB (MPa)
!G_B_yz6                !Shear Modulus of PWB (MPa)
!V_B_yz                 !Poisson's Ratio of PWB
!CTE_B_x                !CTE of PWB (mm/mmK)
!! board y direction
!E_B_y1                 !Young's Modulus of PWB (MPa)
!E_B_y2                 !Young's Modulus of PWB (MPa)
!E_B_y3                 !Young's Modulus of PWB (MPa)
!E_B_y4                 !Young's Modulus of PWB (MPa)
!E_B_y5                 !Young's Modulus of PWB (MPa)
!E_B_y6                 !Young's Modulus of PWB (MPa)
!G_B_zx1                !Shear Modulus of PWB (MPa)
!G_B_zx2                !Shear Modulus of PWB (MPa)
!G_B_zx3                !Shear Modulus of PWB (MPa)
!G_B_zx4                !Shear Modulus of PWB (MPa)
!G_B_zx5                !Shear Modulus of PWB (MPa)
!G_B_zx6                !Shear Modulus of PWB (MPa)
!V_B_zx =               !Poisson's Ratio of PWB
```

```
!CTE_B_y1               !CTE of PWB (mm/mmK)
!CTE_B_y2               !CTE of PWB (mm/mmK)
!CTE_B_y3               !CTE of PWB (mm/mmK)
!CTE_B_y4               !CTE of PWB (mm/mmK)
!CTE_B_y5               !CTE of PWB (mm/mmK)
!CTE_B_y6               !CTE of PWB (mm/mmK)
!! board z direction
!E_B_z1                 !Young's Modulus of PWB (MPa)
!E_B_z2                 !Young's Modulus of PWB (MPa)
!E_B_z3                 !Young's Modulus of PWB (MPa)
!E_B_z4                 !Young's Modulus of PWB (MPa)
!E_B_z5                 !Young's Modulus of PWB (MPa)
!E_B_z6                 !Young's Modulus of PWB (MPa)
!G_B_xy1                !Shear Modulus of PWB (MPa)
!G_B_xy2                !Shear Modulus of PWB (MPa)
!G_B_xy3                !Shear Modulus of PWB (MPa)
!G_B_xy4                !Shear Modulus of PWB (MPa)
!G_B_xy5                !Shear Modulus of PWB (MPa)
!G_B_xy6                !Shear Modulus of PWB (MPa)
!V_B_xy                 !Poisson's Ratio of PWB
!CTE_B_z                !CTE of PWB (mm/mmK)

!! solder ball
!62Sn36Pb2Ag Solder
!M_SJ_T1                !Temperature (deg K)
!M_SJ_T2                !Temperature (deg K)
!M_SJ_T3                !Temperature (deg K)
!M_SJ_T4                !Temperature (deg K)
!M_SJ_T5                !Temperature (deg K)
!E_SJ_x1                !Young's Modulus of Solder (MPa)
!E_SJ_x2                !Young's Modulus of Solder (MPa)
!E_SJ_x3                !Young's Modulus of Solder (MPa)
!E_SJ_x4                !Young's Modulus of Solder (MPa)
!E_SJ_x5                !Young's Modulus of Solder (MPa)
!V_SJ                    !Poisson's Ratio of Solder
!CTE_SJ                  !CTE of Solder (mm/mmK)
! (1)So (2)Q/R (3)A (4)Xi (5)m (6)Ho (7)Sh (8)n (9)a
!So_SJ                  !initial value of deformation resistance (MPa)
!QR_SJ                  !Q:activation energy and R:universal gas constant (1/K)
!A_Const_SJ             !pre-exponential factor (1/sec)
!Xi_SJ                  !multiplier of stress
!m_SJ                    !strain rate sensitivity of stress
!Ho_SJ                  !hardening / softening constant (MPa)
!Sh_SJ                  !coefficient for deformation resistance saturation value (MPa)
!n_SJ                    !strain rate sensitivity of saturation (deformation resistance) value
!a_SJ                    !strain rate sensitivity of hardening or softening

!! copper-pad
!E_PAD                  !Young's Modulus of Component (MPa)
!V_PAD                  !Poisson's Ratio of Component
!G_PAD                   !Shear Modulus (MPa)
!CTE_PAD                 !CTE of Component (mm/mmK)
!STRA_PAD_1              !Strain of Component (PTH)
!STRA_PAD_2              !Strain of Component (PTH)
!STRA_PAD_3              !Strain of Component (PTH)
!STRA_PAD_4              !Strain of Component (PTH)
!STRA_PAD_5              !Strain of Component (PTH)
!STRE_PAD_1              !Stress of Component (PTH)
!STRE_PAD_2              !Stress of Component (PTH)
!STRE_PAD_3              !Stress of Component (PTH)
!STRE_PAD_4              !Stress of Component (PTH)
!STRE_PAD_5              !Stress of Component (PTH)
```

!! Geometric Parameters
L_CBGA = 32.5
Eff_L_CBGA = 45.96194077712559
Th_CBGA = 2.9
Th_PWB = 2.8
P_sb = 1.27
Eff_P_sb = 1.796
Pos_First_sb = 1.796
N_sb = 625.0
Eff_N_sb = 12.0
D_sb = 0.89
H_sb = 0.854
D_b_pad = 0.72
D_c_pad = 0.86
Th_pad = 0.018
!! Operating Parameters
T_0 = 20.0 + 273
T_C0 = 20.0 + 273
T_C1 = -15.0 + 273
T_C2 = 70.0 + 273
T_B0 = 20.0 + 273
T_B1 = -15.0 + 273
T_B2 = 70.0 + 273
time_RMP = 1.0
time_DHI = 10.0
time_DLO = 36.0
!! Material Parameters
!! Component
E_C = 255110.0
V_C = 0.3
G_C = 98119.0
CTE_C = 6.7E-6
!! board(FR4)
M_B_T1 = 303.0
M_B_T2 = 368.0
M_B_T3 = 383.0
M_B_T4 = 398.0
M_B_T5 = 423.0
M_B_T6 = 543.0
E_B_x1 = 22400.0
E_B_x2 = 20680.0
E_B_x3 = 19970.0
E_B_x4 = 19300.0
E_B_x5 = 17920.0
E_B_x6 = 16000.0
G_B_yz1 = 199.0
G_B_yz2 = 189.0
G_B_yz3 = 173.0
G_B_yz4 = 157.0
G_B_yz5 = 142.0
G_B_yz6 = 139.3
V_B_yz = 0.1425
CTE_B_x = 2.0E-5
E_B_y1 = 1600.0
E_B_y2 = 1200.0
E_B_y3 = 1100.0
E_B_y4 = 1000.0
E_B_y5 = 600.0
E_B_y6 = 450.0
G_B_zx1 = 630.0
G_B_zx2 = 600.0

G_B_zx3 = 550.0
G_B_zx4 = 500.0
G_B_zx5 = 450.0
G_B_zx6 = 441.0
V_B_zx = 0.136
CTE_B_y1 = 8.65E-5
CTE_B_y2 = 8.65E-5
CTE_B_y3 = 2.43E-4
CTE_B_y4 = 4.0E-4
CTE_B_y5 = 4.0E-4
CTE_B_y6 = 4.0E-4
E_B_z1 = 22400.0
E_B_z2 = 20680.0
E_B_z3 = 19970.0
E_B_z4 = 19300.0
E_B_z5 = 17920.0
E_B_z6 = 16000.0
G_B_xy1 = 199.0
G_B_xy2 = 189.0
G_B_xy3 = 173.0
G_B_xy4 = 157.0
G_B_xy5 = 142.0
G_B_xy6 = 139.3
V_B_xy = 0.1425
CTE_B_z = 2.0E-5
!! solder joint
M_SJ_T1 = 218.0
M_SJ_T2 = 248.0
M_SJ_T3 = 298.0
M_SJ_T4 = 385.0
M_SJ_T5 = 398.0
E_SJ_x1 = 43400.0
E_SJ_x2 = 38880.0
E_SJ_x3 = 31300.0
E_SJ_x4 = 22200.0
E_SJ_x5 = 16200.0
V_SJ = 0.35
CTE_SJ = 2.4500000000000006E-5
So_SJ = 12.41
QR_SJ = 9400.0
A_Const_SJ = 4000000.0
Xi_SJ = 1.5
m_SJ = 0.303
Ho_SJ = 1379.0
Sh_SJ = 13.79
n_SJ = 0.07
a_SJ = 1.3
!! pad(copper)
E_PAD = 120000.0
V_PAD = 0.3
G_PAD = 448.0
CTE_PAD = 1.7E-5
STRA_PAD_1 = 3.0E-4
STRA_PAD_2 = 0.0010
STRA_PAD_3 = 0.0040
STRA_PAD_4 = 0.01
STRA_PAD_5 = 0.02
STRE_PAD_1 = 30.0
STRE_PAD_2 = 110.0
STRE_PAD_3 = 186.0
STRE_PAD_4 = 217.0
STRE_PAD_5 = 234.0

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.2 define element types

ET,1,Solid45,,,,,
ET,2,VISCO107,,,,,


!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.3 create Material models

!! CBGA
UIMP,CBGA,EX,EY,EZ,E_C,E_C,E_C
UIMP,CBGA,NUXZ,NUXY,NUYZ,V_C,V_C,V_C
UIMP,CBGA,ALPX,ALPY,ALPZ,CTE_C,CTE_C,CTE_C
UIMP,CBGA,GXZ,GXY,GYZ,E_C/(2+2*V_C),E_C/(2+2*V_C),E_C/(2+2*V_C)
UIMP,CBGA,REFT,,,T_0
UIMP,CBGA,DENS,,,1 !dummy value

!! Board
MPTEMP         ! Clear material temperature table
MPTEMP,1,M_B_T1,M_B_T2,M_B_T3,M_B_T4,M_B_T5,M_B_T6
! Young's Modulus
MPDATA,EX,board,1,E_B_x1,E_B_x2,E_B_x3,E_B_x4,E_B_x5,E_B_x6
MPDATA,EY,board,1,E_B_y1,E_B_y2,E_B_y3,E_B_y4,E_B_y5,E_B_y6
MPDATA,EZ,board,1,E_B_z1,E_B_z2,E_B_z3,E_B_z4,E_B_z5,E_B_z6
! SHEAR MODULUS
MPDATA,GYZ,board,1,G_B_yz1,G_B_yz2,G_B_yz3,G_B_yz4,G_B_yz5,G_B_yz6
MPDATA,GXZ,board,1,G_B_zx1,G_B_zx2,G_B_zx3,G_B_zx4,G_B_zx5,G_B_zx6
MPDATA,GXY,board,1,G_B_xy1,G_B_xy2,G_B_xy3,G_B_xy4,G_B_xy5,G_B_xy6
! Poisson's Ratio
MPDATA,NUYZ,board,1,V_B_yz,V_B_yz,V_B_yz,V_B_yz,V_B_yz,V_B_yz
MPDATA,NUXZ,board,1,V_B_zx,V_B_zx,V_B_zx,V_B_zx,V_B_zx,V_B_zx
MPDATA,NUXY,board,1,V_B_xy,V_B_xy,V_B_xy,V_B_xy,V_B_xy,V_B_xy
! Coefficient of Thermal Expansion
MPDATA,ALPX,board,1,CTE_B_x,CTE_B_x,CTE_B_x,CTE_B_x,CTE_B_x,CTE_B_x
MPDATA,ALPY,board,1,CTE_B_y1,CTE_B_y2,CTE_B_y3,CTE_B_y4,CTE_B_y5,CTE_B_y6
MPDATA,ALPZ,board,1,CTE_B_z,CTE_B_z,CTE_B_z,CTE_B_z,CTE_B_z,CTE_B_z
UIMP,board,DENS,,,1 !dummy value

!! Solder Joint
!62Sn36Pb2Ag Solder (Viscoplastic model using Anands)
MPTEMP         ! Clear material temperature table
!MPTEMP,1,M_SJ_T1,M_SJ_T2,M_SJ_T3,M_SJ_T4,M_SJ_T5,M_SJ_T6
! Young's modulus (N/mm^2)
!MPDATA,EX,solder_ball,1,E_SJ_x1,E_SJ_x2,E_SJ_x3,E_SJ_x4,E_SJ_x5,E_SJ_x6
MPTEMP,1,M_SJ_T1,M_SJ_T2,M_SJ_T3,M_SJ_T4,M_SJ_T5
MPDATA,EX,solder_ball,1,E_SJ_x1,E_SJ_x2,E_SJ_x3,E_SJ_x4,E_SJ_x5 ! Young's modulus (N/mm^2)
MPDATA,EY,solder_ball,1,E_SJ_x1,E_SJ_x2,E_SJ_x3,E_SJ_x4,E_SJ_x5 ! Young's modulus  (N/mm^2)
MPDATA,EZ,solder_ball,1,E_SJ_x1,E_SJ_x2,E_SJ_x3,E_SJ_x4,E_SJ_x5 ! Young's modulus  (N/mm^2)
!MPDATA,GYZ,solder_ball,1,G_SJ_x1,G_SJ_x2,G_SJ_x3,G_SJ_x4,G_SJ_x5 ! Shear modulus (N/mm^2)
!MPDATA,GXZ,solder_ball,1,G_SJ_x1,G_SJ_x2,G_SJ_x3,G_SJ_x4,G_SJ_x5 ! Shear modulus  (N/mm^2)
!MPDATA,GXY,solder_ball,1,G_SJ_x1,G_SJ_x2,G_SJ_x3,G_SJ_x4,G_SJ_x5 ! Shear modulus  (N/mm^2)
MP,ALPX,solder_ball,CTE_SJ          ! Coefficient of thermal expansion (mm/mm C)
MP,ALPY,solder_ball,CTE_SJ          ! Coefficient of thermal expansion (mm/mm C)
MP,ALPZ,solder_ball,CTE_SJ          ! Coefficient of thermal expansion (mm/mm C)
MP,NUYZ,solder_ball,V_SJ            ! Poisson's ratio
MP,NUXZ,solder_ball,V_SJ            ! Poisson's ratio
MP,NUXY,solder_ball,V_SJ            ! Poisson's ratio
TB,ANAND, solder_ball
! (1)So  (2)Q/R  (3)A  (4)Xi  (5)m  (6)Ho  (7)Sh  (8)n  (9)a
TBDATA,1, So_SJ, QR_S J, A_Const_SJ, Xi_SJ, m_SJ, Ho_SJ
```

```
TBDATA,7, Sh_SJ, n_SJ, a_SJ
UIMP,solder_ball,DENS,,,1 !dummy value

!! copper-pad
UIMP,pad,EX,EY,EZ,E_PAD,E_PAD,E_PAD
UIMP,pad,NUYZ,NUXZ,NUXY,V_PAD,V_PAD,V_PAD
UIMP,pad,ALPX,ALPY,ALPZ,CTE_PAD,CTE_PAD,CTE_PAD
UIMP,pad,GYZ,GXZ,GXY,G_PAD,G_PAD,G_PAD
UIMP,pad,REFT,,,T_0
UIMP,pad,DENS,,,1 !dummy value
TB, MKIN, pad,              ! MKIN table for copper Strain (mm/mm), stress (N/mm^2)
TBTEMP, , STRAIN
TBDATA, 1, STRA_PAD_1, STRA_PAD_2, STRA_PAD_3, STRA_PAD_4, STRA_PAD_5
TBTEMP
TBDATA, 1, STRE_PAD_1, STRE_PAD_2, STRE_PAD_3, STRE_PAD_4, STRE_PAD_5

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.4 build geometry

!!Left bottom corner is origin
!! Geometric Parameters
D_pad = (D_b_pad + D_c_pad) / 2.0

!! Key points
!! for solder balls and pads

*if,H_sb,le,D_sb,then
        *do,i,1,Eff_N_sb,1
        !position values
        sb_a_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
        sb_a_y%i%=Th_PWB
        sb_a_z%i%=Eff_P_sb/2
        sb_b_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
        sb_b_y%i%=Th_PWB
        sb_b_z%i%=D_pad/2+Eff_P_sb/2
        sb_c_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
        sb_c_y%i%=Th_PWB + Th_pad
        sb_c_z%i%=Eff_P_sb/2
        sb_d_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
        sb_d_y%i%=Th_PWB + Th_pad
        sb_d_z%i%=D_pad/2+Eff_P_sb/2
        sb_e_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
        sb_e_y%i%=Th_PWB + Th_pad + H_sb*0.2
        sb_e_z%i%=Eff_P_sb/2
        sb_f_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
        sb_f_y%i%=Th_PWB + Th_pad + H_sb*0.2
        sb_f_z%i%=D_pad/2+Eff_P_sb/2
        sb_g_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
        sb_g_y%i%=Th_PWB + Th_pad + H_sb*0.2
        sb_g_z%i%=D_sb/2 + Eff_P_sb/2
        sb_h_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
        sb_h_y%i%=Th_PWB + Th_pad + H_sb*0.8
        sb_h_z%i%=Eff_P_sb/2
        sb_i_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
        sb_i_y%i%=Th_PWB + Th_pad + H_sb*0.8
        sb_i_z%i%=D_pad/2+Eff_P_sb/2
        sb_j_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
        sb_j_y%i%=Th_PWB + Th_pad + H_sb*0.8
        sb_j_z%i%=D_sb/2 + Eff_P_sb/2
        sb_k_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
        sb_k_y%i%=Th_PWB + Th_pad + H_sb
        sb_k_z%i%=Eff_P_sb/2
```

```
        sb_l_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
        sb_l_y%i%=Th_PWB + Th_pad + H_sb
        sb_l_z%i%=D_pad/2+Eff_P_sb/2
        sb_m_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
        sb_m_y%i%=Th_PWB + Th_pad + H_sb + Th_pad
        sb_m_z%i%=Eff_P_sb/2
        sb_n_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
        sb_n_y%i%=Th_PWB + Th_pad + H_sb + Th_pad
        sb_n_z%i%=D_pad/2+Eff_P_sb/2
        !points a to n
        k,i*14-13,sb_a_x%i%,sb_a_y%i%,sb_a_z%i%
        k,i*14-12,sb_b_x%i%,sb_b_y%i%,sb_b_z%i%
        k,i*14-11,sb_c_x%i%,sb_c_y%i%,sb_c_z%i%
        k,i*14-10,sb_d_x%i%,sb_d_y%i%,sb_d_z%i%
        k,i*14-9,sb_e_x%i%,sb_e_y%i%,sb_e_z%i%
        k,i*14-8,sb_f_x%i%,sb_f_y%i%,sb_f_z%i%
        k,i*14-7,sb_g_x%i%,sb_g_y%i%,sb_g_z%i%
        k,i*14-6,sb_h_x%i%,sb_h_y%i%,sb_h_z%i%
        k,i*14-5,sb_i_x%i%,sb_i_y%i%,sb_i_z%i%
        k,i*14-4,sb_j_x%i%,sb_j_y%i%,sb_j_z%i%
        k,i*14-3,sb_k_x%i%,sb_k_y%i%,sb_k_z%i%
        k,i*14-2,sb_l_x%i%,sb_l_y%i%,sb_l_z%i%
        k,i*14-1,sb_m_x%i%,sb_m_y%i%,sb_m_z%i%
        k,i*14,sb_n_x%i%,sb_n_y%i%,sb_n_z%i%
        ! areas 1,2,3,4,5,6,7,8
        a,i*14-13,i*14-12,i*14-10,i*14-11
        a,i*14-11,i*14-10,i*14-8,i*14-9
        a,i*14-10,i*14-7,i*14-8
        a,i*14-9,i*14-8,i*14-5,i*14-6
        a,i*14-8,i*14-7,i*14-4,i*14-5
        a,i*14-6,i*14-5,i*14-2,i*14-3
        a,i*14-5,i*14-4,i*14-2
        a,i*14-3,i*14-2,i*14,i*14-1
        *enddo

        ! volumes
        *do,i,1,Eff_N_sb,1
        vrotat,i*8-7,,,,,,i*14-13,i*14-1,360,4,
        vrotat,i*8-6,,,,,,i*14-13,i*14-1,360,4,
        vrotat,i*8-5,,,,,,i*14-13,i*14-1,360,4,
        vrotat,i*8-4,,,,,,i*14-13,i*14-1,360,4,
        vrotat,i*8-3,,,,,,i*14-13,i*14-1,360,4,
        vrotat,i*8-2,,,,,,i*14-13,i*14-1,360,4,
        vrotat,i*8-1,,,,,,i*14-13,i*14-1,360,4,
        vrotat,i*8,,,,,,i*14-13,i*14-1,360,4,
        *enddo
        !! Geometric Parameters
        P_column = D_sb

*else
        *do,i,1,Eff_N_sb,1
        !position values
        sb_a_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
        sb_a_y%i%=Th_PWB
        sb_a_z%i%=Eff_P_sb/2
        sb_b_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
        sb_b_y%i%=Th_PWB
        sb_b_z%i%=D_sb/2+Eff_P_sb/2
        sb_c_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
        sb_c_y%i%=Th_PWB
        sb_c_z%i%=D_pad/2+Eff_P_sb/2
        sb_d_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
```

```
sb_d_y%i%=Th_PWB + Th_pad
sb_d_z%i%=Eff_P_sb/2
sb_e_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
sb_e_y%i%=Th_PWB + Th_pad
sb_e_z%i%=D_sb/2+Eff_P_sb/2
sb_f_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
sb_f_y%i%=Th_PWB + Th_pad
sb_f_z%i%=D_pad/2+Eff_P_sb/2
sb_g_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
sb_g_y%i%=Th_PWB + Th_pad + H_sb*0.2
sb_g_z%i%=Eff_P_sb/2
sb_h_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
sb_h_y%i%=Th_PWB + Th_pad + H_sb*0.2
sb_h_z%i%=D_sb/2+Eff_P_sb/2
sb_i_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
sb_i_y%i%=Th_PWB + Th_pad + H_sb*0.8
sb_i_z%i%=Eff_P_sb/2
sb_j_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
sb_j_y%i%=Th_PWB + Th_pad + H_sb*0.8
sb_j_z%i%=D_sb/2 + Eff_P_sb/2
sb_k_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
sb_k_y%i%=Th_PWB + Th_pad + H_sb
sb_k_z%i%=Eff_P_sb/2
sb_l_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
sb_l_y%i%=Th_PWB + Th_pad + H_sb
sb_l_z%i%=D_sb/2+Eff_P_sb/2
sb_m_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
sb_m_y%i%=Th_PWB + Th_pad + H_sb
sb_m_z%i%=D_pad/2+Eff_P_sb/2
sb_n_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
sb_n_y%i%=Th_PWB + Th_pad + H_sb + Th_pad
sb_n_z%i%=Eff_P_sb/2
sb_o_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
sb_o_y%i%=Th_PWB + Th_pad + H_sb + Th_pad
sb_o_z%i%=D_sb/2+Eff_P_sb/2
sb_p_x%i%=Pos_First_sb + Eff_P_sb*(i-1)
sb_p_y%i%=Th_PWB + Th_pad + H_sb + Th_pad
sb_p_z%i%=D_pad/2+Eff_P_sb/2
!points a to n
k,i*16-15,sb_a_x%i%,sb_a_y%i%,sb_a_z%i%
k,i*16-14,sb_b_x%i%,sb_b_y%i%,sb_b_z%i%
k,i*16-13,sb_c_x%i%,sb_c_y%i%,sb_c_z%i%
k,i*16-12,sb_d_x%i%,sb_d_y%i%,sb_d_z%i%
k,i*16-11,sb_e_x%i%,sb_e_y%i%,sb_e_z%i%
k,i*16-10,sb_f_x%i%,sb_f_y%i%,sb_f_z%i%
k,i*16-9,sb_g_x%i%,sb_g_y%i%,sb_g_z%i%
k,i*16-8,sb_h_x%i%,sb_h_y%i%,sb_h_z%i%
k,i*16-7,sb_i_x%i%,sb_i_y%i%,sb_i_z%i%
k,i*16-6,sb_j_x%i%,sb_j_y%i%,sb_j_z%i%
k,i*16-5,sb_k_x%i%,sb_k_y%i%,sb_k_z%i%
k,i*16-4,sb_l_x%i%,sb_l_y%i%,sb_l_z%i%
k,i*16-3,sb_m_x%i%,sb_m_y%i%,sb_m_z%i%
k,i*16-2,sb_n_x%i%,sb_n_y%i%,sb_n_z%i%
k,i*16-1,sb_o_x%i%,sb_o_y%i%,sb_o_z%i%
k,i*16,sb_p_x%i%,sb_p_y%i%,sb_p_z%i%
! areas 1,2,3,4,5,6,7,8,9
a,i*16-15,i*16-14,i*16-11,i*16-12
a,i*16-14,i*16-13,i*16-10,i*16-11
a,i*16-12,i*16-11,i*16-8,i*16-9
a,i*16-11,i*16-10,i*16-8
a,i*16-9,i*16-8,i*16-6,i*16-7
a,i*16-7,i*16-6,i*16-4,i*16-5
```

```
        a,i*16-6,i*16-3,i*16-4
        a,i*16-5,i*16-4,i*16-1,i*16-2
        a,i*16-4,i*16-3,i*16,i*16-1
        *enddo

        ! volumes
        *do,i,1,Eff_N_sb,1
        vrotat,i*9-8,,,,,,i*16-15,i*16-2,360,4,
        vrotat,i*9-7,,,,,,i*16-15,i*16-2,360,4,
        vrotat,i*9-6,,,,,,i*16-15,i*16-2,360,4,
        vrotat,i*9-5,,,,,,i*16-15,i*16-2,360,4,
        vrotat,i*9-4,,,,,,i*16-15,i*16-2,360,4,
        vrotat,i*9-3,,,,,,i*16-15,i*16-2,360,4,
        vrotat,i*9-2,,,,,,i*16-15,i*16-2,360,4,
        vrotat,i*9-1,,,,,,i*16-15,i*16-2,360,4,
        vrotat,i*9,,,,,,i*16-15,i*16-2,360,4,
        *enddo
        !! Geometric Parameters
        P_column = D_pad*1.2
*endif

*do,i,1,Eff_N_sb,1
pwb_a_x%i%=Pos_First_sb + Eff_P_sb*(i-1) - P_column/2
pwb_a_y%i%=0
pwb_a_z%i%=0
pwb_b_x%i%=Pos_First_sb + Eff_P_sb*(i-1) - P_column/2
pwb_b_y%i%=0
pwb_b_z%i%=Eff_P_sb
pwb_c_x%i%=Pos_First_sb + Eff_P_sb*(i-1) + P_column/2
pwb_c_y%i%=0
pwb_c_z%i%=Eff_P_sb
pwb_d_x%i%=Pos_First_sb + Eff_P_sb*(i-1) + P_column/2
pwb_d_y%i%=0
pwb_d_z%i%=0
pwb_e_x%i%=Pos_First_sb + Eff_P_sb*(i-1) - P_column/2
pwb_e_y%i%=Th_PWB
pwb_e_z%i%=0
pwb_f_x%i%=Pos_First_sb + Eff_P_sb*(i-1) - P_column/2
pwb_f_y%i%=Th_PWB
pwb_f_z%i%=Eff_P_sb
pwb_g_x%i%=Pos_First_sb + Eff_P_sb*(i-1) + P_column/2
pwb_g_y%i%=Th_PWB
pwb_g_z%i%=Eff_P_sb
pwb_h_x%i%=Pos_First_sb + Eff_P_sb*(i-1) + P_column/2
pwb_h_y%i%=Th_PWB
pwb_h_z%i%=0
k,2000+i*8-7,pwb_a_x%i%,pwb_a_y%i%,pwb_a_z%i%
k,2000+i*8-6,pwb_b_x%i%,pwb_b_y%i%,pwb_b_z%i%
k,2000+i*8-5,pwb_c_x%i%,pwb_c_y%i%,pwb_c_z%i%
k,2000+i*8-4,pwb_d_x%i%,pwb_d_y%i%,pwb_d_z%i%
k,2000+i*8-3,pwb_e_x%i%,pwb_e_y%i%,pwb_e_z%i%
k,2000+i*8-2,pwb_f_x%i%,pwb_f_y%i%,pwb_f_z%i%
k,2000+i*8-1,pwb_g_x%i%,pwb_g_y%i%,pwb_g_z%i%
k,2000+i*8,pwb_h_x%i%,pwb_h_y%i%,pwb_h_z%i%
*enddo

*do,i,1,Eff_N_sb,1
v,2000+i*8-7,2000+i*8-6,2000+i*8-5,2000+i*8-4,2000+i*8-3,2000+i*8-2,2000+i*8-1,2000+i*8
*enddo

*do,i,1,Eff_N_sb,1
cbga_a_x%i%=Pos_First_sb + Eff_P_sb*(i-1) - P_column/2
```

```
cbga_a_y%i%=Th_PWB + Th_pad + H_sb + Th_pad
cbga_a_z%i%=0
cbga_b_x%i%=Pos_First_sb + Eff_P_sb*(i-1) - P_column/2
cbga_b_y%i%=Th_PWB + Th_pad + H_sb + Th_pad
cbga_b_z%i%=Eff_P_sb
cbga_c_x%i%=Pos_First_sb + Eff_P_sb*(i-1) + P_column/2
cbga_c_y%i%=Th_PWB + Th_pad + H_sb + Th_pad
cbga_c_z%i%=Eff_P_sb
cbga_d_x%i%=Pos_First_sb + Eff_P_sb*(i-1) + P_column/2
cbga_d_y%i%=Th_PWB + Th_pad + H_sb + Th_pad
cbga_d_z%i%=0
cbga_e_x%i%=Pos_First_sb + Eff_P_sb*(i-1) - P_column/2
cbga_e_y%i%=Th_PWB + Th_pad + H_sb + Th_pad + Th_CBGA
cbga_e_z%i%=0
cbga_f_x%i%=Pos_First_sb + Eff_P_sb*(i-1) - P_column/2
cbga_f_y%i%=Th_PWB + Th_pad + H_sb + Th_pad + Th_CBGA
cbga_f_z%i%=Eff_P_sb
cbga_g_x%i%=Pos_First_sb + Eff_P_sb*(i-1) + P_column/2
cbga_g_y%i%=Th_PWB + Th_pad + H_sb + Th_pad + Th_CBGA
cbga_g_z%i%=Eff_P_sb
cbga_h_x%i%=Pos_First_sb + Eff_P_sb*(i-1) + P_column/2
cbga_h_y%i%=Th_PWB + Th_pad + H_sb + Th_pad + Th_CBGA
cbga_h_z%i%=0
k,3000+i*8-7,cbga_a_x%i%,cbga_a_y%i%,cbga_a_z%i%
k,3000+i*8-6,cbga_b_x%i%,cbga_b_y%i%,cbga_b_z%i%
k,3000+i*8-5,cbga_c_x%i%,cbga_c_y%i%,cbga_c_z%i%
k,3000+i*8-4,cbga_d_x%i%,cbga_d_y%i%,cbga_d_z%i%
k,3000+i*8-3,cbga_e_x%i%,cbga_e_y%i%,cbga_e_z%i%
k,3000+i*8-2,cbga_f_x%i%,cbga_f_y%i%,cbga_f_z%i%
k,3000+i*8-1,cbga_g_x%i%,cbga_g_y%i%,cbga_g_z%i%
k,3000+i*8,cbga_h_x%i%,cbga_h_y%i%,cbga_h_z%i%
*enddo

*do,i,1,Eff_N_sb,1
v,3000+i*8-7,3000+i*8-6,3000+i*8-5,3000+i*8-4,3000+i*8-3,3000+i*8-2,3000+i*8-1,3000+i*8
*enddo

*do,i,1,Eff_N_sb-1,1
pwb_a2_x%i%=Pos_First_sb + Eff_P_sb*(i-1) + P_column/2
pwb_a2_y%i%=0
pwb_a2_z%i%=0
pwb_b2_x%i%=Pos_First_sb + Eff_P_sb*(i-1) + P_column/2
pwb_b2_y%i%=0
pwb_b2_z%i%=Eff_P_sb
pwb_c2_x%i%=Pos_First_sb + Eff_P_sb*(i) - P_column/2
pwb_c2_y%i%=0
pwb_c2_z%i%=Eff_P_sb
pwb_d2_x%i%=Pos_First_sb + Eff_P_sb*(i) - P_column/2
pwb_d2_y%i%=0
pwb_d2_z%i%=0
pwb_e2_x%i%=Pos_First_sb + Eff_P_sb*(i-1) + P_column/2
pwb_e2_y%i%=Th_PWB
pwb_e2_z%i%=0
pwb_f2_x%i%=Pos_First_sb + Eff_P_sb*(i-1) + P_column/2
pwb_f2_y%i%=Th_PWB
pwb_f2_z%i%=Eff_P_sb
pwb_g2_x%i%=Pos_First_sb + Eff_P_sb*(i) - P_column/2
pwb_g2_y%i%=Th_PWB
pwb_g2_z%i%=Eff_P_sb
pwb_h2_x%i%=Pos_First_sb + Eff_P_sb*(i) - P_column/2
pwb_h2_y%i%=Th_PWB
pwb_h2_z%i%=0
```

```
k,4000+i*8-7,pwb_a2_x%i%,pwb_a2_y%i%,pwb_a2_z%i%
k,4000+i*8-6,pwb_b2_x%i%,pwb_b2_y%i%,pwb_b2_z%i%
k,4000+i*8-5,pwb_c2_x%i%,pwb_c2_y%i%,pwb_c2_z%i%
k,4000+i*8-4,pwb_d2_x%i%,pwb_d2_y%i%,pwb_d2_z%i%
k,4000+i*8-3,pwb_e2_x%i%,pwb_e2_y%i%,pwb_e2_z%i%
k,4000+i*8-2,pwb_f2_x%i%,pwb_f2_y%i%,pwb_f2_z%i%
k,4000+i*8-1,pwb_g2_x%i%,pwb_g2_y%i%,pwb_g2_z%i%
k,4000+i*8,pwb_h2_x%i%,pwb_h2_y%i%,pwb_h2_z%i%
*enddo

*do,i,1,Eff_N_sb-1,1
v,4000+i*8-7,4000+i*8-6,4000+i*8-5,4000+i*8-4,4000+i*8-3,4000+i*8-2,4000+i*8-1,4000+i*8
*enddo

*do,i,1,Eff_N_sb-1,1
cbga_a2_x%i%=Pos_First_sb + Eff_P_sb*(i-1) + P_column/2
cbga_a2_y%i%=Th_PWB + Th_pad + H_sb + Th_pad
cbga_a2_z%i%=0
cbga_b2_x%i%=Pos_First_sb + Eff_P_sb*(i-1) + P_column/2
cbga_b2_y%i%=Th_PWB + Th_pad + H_sb + Th_pad
cbga_b2_z%i%=Eff_P_sb
cbga_c2_x%i%=Pos_First_sb + Eff_P_sb*(i) - P_column/2
cbga_c2_y%i%=Th_PWB + Th_pad + H_sb + Th_pad
cbga_c2_z%i%=Eff_P_sb
cbga_d2_x%i%=Pos_First_sb + Eff_P_sb*(i) - P_column/2
cbga_d2_y%i%=Th_PWB + Th_pad + H_sb + Th_pad
cbga_d2_z%i%=0
cbga_e2_x%i%=Pos_First_sb + Eff_P_sb*(i-1) + P_column/2
cbga_e2_y%i%=Th_PWB + Th_pad + H_sb + Th_pad + Th_CBGA
cbga_e2_z%i%=0
cbga_f2_x%i%=Pos_First_sb + Eff_P_sb*(i-1) + P_column/2
cbga_f2_y%i%=Th_PWB + Th_pad + H_sb + Th_pad + Th_CBGA
cbga_f2_z%i%=Eff_P_sb
cbga_g2_x%i%=Pos_First_sb + Eff_P_sb*(i) - P_column/2
cbga_g2_y%i%=Th_PWB + Th_pad + H_sb + Th_pad + Th_CBGA
cbga_g2_z%i%=Eff_P_sb
cbga_h2_x%i%=Pos_First_sb + Eff_P_sb*(i) - P_column/2
cbga_h2_y%i%=Th_PWB + Th_pad + H_sb + Th_pad + Th_CBGA
cbga_h2_z%i%=0
k,5000+i*8-7,cbga_a2_x%i%,cbga_a2_y%i%,cbga_a2_z%i%
k,5000+i*8-6,cbga_b2_x%i%,cbga_b2_y%i%,cbga_b2_z%i%
k,5000+i*8-5,cbga_c2_x%i%,cbga_c2_y%i%,cbga_c2_z%i%
k,5000+i*8-4,cbga_d2_x%i%,cbga_d2_y%i%,cbga_d2_z%i%
k,5000+i*8-3,cbga_e2_x%i%,cbga_e2_y%i%,cbga_e2_z%i%
k,5000+i*8-2,cbga_f2_x%i%,cbga_f2_y%i%,cbga_f2_z%i%
k,5000+i*8-1,cbga_g2_x%i%,cbga_g2_y%i%,cbga_g2_z%i%
k,5000+i*8,cbga_h2_x%i%,cbga_h2_y%i%,cbga_h2_z%i%
*enddo

*do,i,1,Eff_N_sb-1,1
v,5000+i*8-7,5000+i*8-6,5000+i*8-5,5000+i*8-4,5000+i*8-3,5000+i*8-2,5000+i*8-1,5000+i*8
*enddo

pwb_a3_x=0
pwb_a3_y=0
pwb_a3_z=0
pwb_b3_x=0
pwb_b3_y=0
pwb_b3_z=Eff_P_sb
pwb_c3_x=Pos_First_sb - P_column/2
pwb_c3_y=0
pwb_c3_z=Eff_P_sb
```

```
pwb_d3_x=Pos_First_sb - P_column/2
pwb_d3_y=0
pwb_d3_z=0
pwb_e3_x=0
pwb_e3_y=Th_PWB
pwb_e3_z=0
pwb_f3_x=0
pwb_f3_y=Th_PWB
pwb_f3_z=Eff_P_sb
pwb_g3_x=Pos_First_sb - P_column/2
pwb_g3_y=Th_PWB
pwb_g3_z=Eff_P_sb
pwb_h3_x=Pos_First_sb - P_column/2
pwb_h3_y=Th_PWB
pwb_h3_z=0
k,6000+1,pwb_a3_x,pwb_a3_y,pwb_a3_z
k,6000+2,pwb_b3_x,pwb_b3_y,pwb_b3_z
k,6000+3,pwb_c3_x,pwb_c3_y,pwb_c3_z
k,6000+4,pwb_d3_x,pwb_d3_y,pwb_d3_z
k,6000+5,pwb_e3_x,pwb_e3_y,pwb_e3_z
k,6000+6,pwb_f3_x,pwb_f3_y,pwb_f3_z
k,6000+7,pwb_g3_x,pwb_g3_y,pwb_g3_z
k,6000+8,pwb_h3_x,pwb_h3_y,pwb_h3_z
v,6001,6002,6003,6004,6005,6006,6007,6008
pwb_a4_x=Pos_First_sb + Eff_P_sb*(Eff_N_sb-1) + P_column/2
pwb_a4_y=0
pwb_a4_z=0
pwb_b4_x=Pos_First_sb + Eff_P_sb*(Eff_N_sb-1) + P_column/2
pwb_b4_y=0
pwb_b4_z=Eff_P_sb
pwb_c4_x=Eff_L_CBGA/2 + P_column/2
pwb_c4_y=0
pwb_c4_z=Eff_P_sb
pwb_d4_x=Eff_L_CBGA/2 + P_column/2
pwb_d4_y=0
pwb_d4_z=0
pwb_e4_x=Pos_First_sb + Eff_P_sb*(Eff_N_sb-1) + P_column/2
pwb_e4_y=Th_PWB
pwb_e4_z=0
pwb_f4_x=Pos_First_sb + Eff_P_sb*(Eff_N_sb-1) + P_column/2
pwb_f4_y=Th_PWB
pwb_f4_z=Eff_P_sb
pwb_g4_x=Eff_L_CBGA/2 + P_column/2
pwb_g4_y=Th_PWB
pwb_g4_z=Eff_P_sb
pwb_h4_x=Eff_L_CBGA/2 + P_column/2
pwb_h4_y=Th_PWB
pwb_h4_z=0
k,6000+9,pwb_a4_x,pwb_a4_y,pwb_a4_z
k,6000+10,pwb_b4_x,pwb_b4_y,pwb_b4_z
k,6000+11,pwb_c4_x,pwb_c4_y,pwb_c4_z
k,6000+12,pwb_d4_x,pwb_d4_y,pwb_d4_z
k,6000+13,pwb_e4_x,pwb_e4_y,pwb_e4_z
k,6000+14,pwb_f4_x,pwb_f4_y,pwb_f4_z
k,6000+15,pwb_g4_x,pwb_g4_y,pwb_g4_z
k,6000+16,pwb_h4_x,pwb_h4_y,pwb_h4_z
v,6009,6010,6011,6012,6013,6014,6015,6016
cbga_a3_x=0
cbga_a3_y=Th_PWB + Th_pad + H_sb + Th_pad
cbga_a3_z=0
cbga_b3_x=0
cbga_b3_y=Th_PWB + Th_pad + H_sb + Th_pad
```

```
cbga_b3_z=Eff_P_sb
cbga_c3_x=Pos_First_sb - P_column/2
cbga_c3_y=Th_PWB + Th_pad + H_sb + Th_pad
cbga_c3_z=Eff_P_sb
cbga_d3_x=Pos_First_sb - P_column/2
cbga_d3_y=Th_PWB + Th_pad + H_sb + Th_pad
cbga_d3_z=0
cbga_e3_x=0
cbga_e3_y=Th_PWB + Th_pad + H_sb + Th_pad + Th_CBGA
cbga_e3_z=0
cbga_f3_x=0
cbga_f3_y=Th_PWB + Th_pad + H_sb + Th_pad + Th_CBGA
cbga_f3_z=Eff_P_sb
cbga_g3_x=Pos_First_sb - P_column/2
cbga_g3_y=Th_PWB + Th_pad + H_sb + Th_pad + Th_CBGA
cbga_g3_z=Eff_P_sb
cbga_h3_x=Pos_First_sb - P_column/2
cbga_h3_y=Th_PWB + Th_pad + H_sb + Th_pad + Th_CBGA
cbga_h3_z=0
k,7000+1,cbga_a3_x,cbga_a3_y,cbga_a3_z
k,7000+2,cbga_b3_x,cbga_b3_y,cbga_b3_z
k,7000+3,cbga_c3_x,cbga_c3_y,cbga_c3_z
k,7000+4,cbga_d3_x,cbga_d3_y,cbga_d3_z
k,7000+5,cbga_e3_x,cbga_e3_y,cbga_e3_z
k,7000+6,cbga_f3_x,cbga_f3_y,cbga_f3_z
k,7000+7,cbga_g3_x,cbga_g3_y,cbga_g3_z
k,7000+8,cbga_h3_x,cbga_h3_y,cbga_h3_z
v,7001,7002,7003,7004,7005,7006,7007,7008
cbga_a4_x=Pos_First_sb + Eff_P_sb*(Eff_N_sb-1) + P_column/2
cbga_a4_y=Th_PWB + Th_pad + H_sb + Th_pad
cbga_a4_z=0
cbga_b4_x=Pos_First_sb + Eff_P_sb*(Eff_N_sb-1) + P_column/2
cbga_b4_y=Th_PWB + Th_pad + H_sb + Th_pad
cbga_b4_z=Eff_P_sb
cbga_c4_x=Eff_L_CBGA/2
cbga_c4_y=Th_PWB + Th_pad + H_sb + Th_pad
cbga_c4_z=Eff_P_sb
cbga_d4_x=Eff_L_CBGA/2
cbga_d4_y=Th_PWB + Th_pad + H_sb + Th_pad
cbga_d4_z=0
cbga_e4_x=Pos_First_sb + Eff_P_sb*(Eff_N_sb-1) + P_column/2
cbga_e4_y=Th_PWB + Th_pad + H_sb + Th_pad + Th_CBGA
cbga_e4_z=0
cbga_f4_x=Pos_First_sb + Eff_P_sb*(Eff_N_sb-1) + P_column/2
cbga_f4_y=Th_PWB + Th_pad + H_sb + Th_pad + Th_CBGA
cbga_f4_z=Eff_P_sb
cbga_g4_x=Eff_L_CBGA/2
cbga_g4_y=Th_PWB + Th_pad + H_sb + Th_pad + Th_CBGA
cbga_g4_z=Eff_P_sb
cbga_h4_x=Eff_L_CBGA/2
cbga_h4_y=Th_PWB + Th_pad + H_sb + Th_pad + Th_CBGA
cbga_h4_z=0
k,7000+9,cbga_a4_x,cbga_a4_y,cbga_a4_z
k,7000+10,cbga_b4_x,cbga_b4_y,cbga_b4_z
k,7000+11,cbga_c4_x,cbga_c4_y,cbga_c4_z
k,7000+12,cbga_d4_x,cbga_d4_y,cbga_d4_z
k,7000+13,cbga_e4_x,cbga_e4_y,cbga_e4_z
k,7000+14,cbga_f4_x,cbga_f4_y,cbga_f4_z
k,7000+15,cbga_g4_x,cbga_g4_y,cbga_g4_z
k,7000+16,cbga_h4_x,cbga_h4_y,cbga_h4_z
v,7009,7010,7011,7012,7013,7014,7015,7016
```

```
allsel,all
vglue,all
vplot

!! Assing Material Models to Volume
allsel,all
vsel,s,loc,y,-incc,Th_PWB+incc
vatt,board,,1,0
vsel,s,loc,y,Th_PWB-incc,Th_PWB+Th_pad+incc
vatt,pad,,1,0
vsel,s,loc,y,Th_PWB+Th_pad-incc,Th_PWB+Th_pad+H_sb+incc
vatt,solder_ball,,2,0
vsel,s,loc,y,Th_PWB+Th_pad+H_sb-incc,Th_PWB+Th_pad+H_sb+Th_pad+incc
vatt,pad,,1,0
vsel,s,loc,y,Th_PWB+Th_pad+H_sb+Th_pad-incc,Th_PWB+Th_pad+H_sb+Th_pad+Th_CBGA+incc
vatt,CBGA,,1,0

!! Line Division for Mesh
!division of lines
*do,i,1,Eff_N_sb-1,1
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(i-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(i-1)+Eff_P_sb/2
lsel,r,loc,y,Th_PWB-incc,Th_PWB+incc
lesize,all,,,m_num_sb
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(i-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(i-1)+Eff_P_sb/2
lsel,r,loc,y,Th_PWB+Th_pad-incc,Th_PWB+Th_pad+incc
lesize,all,,,m_num_sb
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(i-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(i-1)+Eff_P_sb/2
lsel,r,loc,y,Th_PWB+Th_pad+H_sb*0.2-incc,Th_PWB+Th_pad+H_sb*0.2+incc
lesize,all,,,m_num_sb
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(i-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(i-1)+Eff_P_sb/2
lsel,r,loc,y,Th_PWB+Th_pad+H_sb*0.8-incc,Th_PWB+Th_pad+H_sb*0.8+incc
lesize,all,,,m_num_sb
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(i-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(i-1)+Eff_P_sb/2
lsel,r,loc,y,Th_PWB+Th_pad+H_sb-incc,Th_PWB+Th_pad+H_sb+incc
lesize,all,,,m_num_sb
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(i-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(i-1)+Eff_P_sb/2
lsel,r,loc,y,Th_PWB+Th_pad+H_sb+Th_pad-incc,Th_PWB+Th_pad+H_sb+Th_pad+incc
lesize,all,,,m_num_sb
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(i-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(i-1)+Eff_P_sb/2
lsel,r,loc,y,Th_PWB-incc,Th_PWB+Th_pad+incc
lesize,all,,,m_num_sb
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(i-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(i-1)+Eff_P_sb/2
lsel,r,loc,y,Th_PWB+Th_pad+H_sb-incc,Th_PWB+Th_pad+H_sb+Th_pad+incc
lesize,all,,,m_num_sb
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(i-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(i-1)+Eff_P_sb/2
lsel,r,loc,y,Th_PWB+Th_pad-incc,Th_PWB+Th_pad+H_sb+incc
lesize,all,,,m_num_sb
*enddo

lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(Eff_N_sb-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(Eff_N_sb-
1)+Eff_P_sb/2
lsel,r,loc,y,Th_PWB-incc,Th_PWB+incc
lesize,all,,,m_num_sb*4
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(Eff_N_sb-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(Eff_N_sb-
1)+Eff_P_sb/2
lsel,r,loc,y,Th_PWB+Th_pad-incc,Th_PWB+Th_pad+incc
lesize,all,,,m_num_sb*6
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(Eff_N_sb-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(Eff_N_sb-
1)+Eff_P_sb/2
lsel,r,loc,y,Th_PWB+Th_pad+H_sb*0.2-incc,Th_PWB+Th_pad+H_sb*0.2+incc
lesize,all,,,m_num_sb*4
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(Eff_N_sb-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(Eff_N_sb-
```

```
1)+Eff_P_sb/2
lsel,r,loc,y,Th_PWB+Th_pad+H_sb*0.8-incc,Th_PWB+Th_pad+H_sb*0.8+incc
lesize,all,,,m_num_sb*4
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(Eff_N_sb-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(Eff_N_sb-
1)+Eff_P_sb/2
lsel,r,loc,y,Th_PWB+Th_pad+H_sb-incc,Th_PWB+Th_pad+H_sb+incc
lesize,all,,,m_num_sb*6
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(Eff_N_sb-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(Eff_N_sb-
1)+Eff_P_sb/2
lsel,r,loc,y,Th_PWB+Th_pad+H_sb+Th_pad-incc,Th_PWB+Th_pad+H_sb+Th_pad+incc
lesize,all,,,m_num_sb*4
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(Eff_N_sb-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(Eff_N_sb-
1)+Eff_P_sb/2
lsel,r,loc,y,Th_PWB-incc,Th_PWB+Th_pad+incc
lesize,all,,,m_num_sb
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(Eff_N_sb-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(Eff_N_sb-
1)+Eff_P_sb/2
lsel,r,loc,y,Th_PWB+Th_pad+H_sb-incc,Th_PWB+Th_pad+H_sb+Th_pad+incc
lesize,all,,,m_num_sb
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(Eff_N_sb-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(Eff_N_sb-
1)+Eff_P_sb/2
lsel,r,loc,y,Th_PWB+Th_pad-incc,Th_PWB+Th_pad+H_sb*0.2+incc
lesize,all,,,m_num_sb*6
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(Eff_N_sb-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(Eff_N_sb-
1)+Eff_P_sb/2
lsel,r,loc,y,Th_PWB+Th_pad+H_sb*0.2-incc,Th_PWB+Th_pad+H_sb*0.8+incc
lesize,all,,,m_num_sb*4
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(Eff_N_sb-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(Eff_N_sb-
1)+Eff_P_sb/2
lsel,r,loc,y,Th_PWB+Th_pad+H_sb*0.8-incc,Th_PWB+Th_pad+H_sb+incc
lesize,all,,,m_num_sb*6

*do,i,1,Eff_N_sb,1
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(i-1) - P_column/2 - incc,Pos_First_sb + Eff_P_sb*(i-1) - P_column/2 +
incc
lesize,all,,,m_num
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(i-1) + P_column/2 - incc,Pos_First_sb + Eff_P_sb*(i-1) + P_column/2 +
incc
lesize,all,,,m_num
*enddo

*do,i,1,Eff_N_sb-1,1
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(i-1) + P_column/2 - incc,Pos_First_sb + Eff_P_sb*(i-1) + P_column/2 +
incc
lesize,all,,,m_num
lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(i) - P_column/2 - incc,Pos_First_sb + Eff_P_sb*(i) - P_column/2 + incc
lesize,all,,,m_num
*enddo

lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(Eff_N_sb-1) + P_column/2-incc,Pos_First_sb + Eff_P_sb*(Eff_N_sb-1) +
P_column/2+incc
lesize,all,,,m_num
lsel,s,loc,x,Eff_L_CBGA/2 + P_column/2 - incc,Eff_L_CBGA/2 + P_column/2 + incc
lesize,all,,,m_num

lsel,s,loc,x,-incc,+incc
lesize,all,,,m_num
lsel,s,loc,x,Pos_First_sb - P_column/2 - incc,Pos_First_sb - P_column/2 + incc
lesize,all,,,m_num

lsel,s,loc,x,Pos_First_sb + Eff_P_sb*(Eff_N_sb-1) + P_column/2-incc,Pos_First_sb + Eff_P_sb*(Eff_N_sb-1) +
P_column/2+incc
```

```
lesize,all,,,m_num
lsel,s,loc,x,Eff_L_CBGA/2 - incc,Eff_L_CBGA/2 + incc
lesize,all,,,m_num
lsel,s,loc,z,-incc,+incc
lesize,all,,,m_num
lsel,s,loc,z,Eff_P_sb-incc,Eff_P_sb+incc
lesize,all,,,m_num

!! Mesh
allsel,all
mshkey,0 ! free mesh
MSHAPE,1,3d
vmesh,all

allsel, all
nummrg,all,1e-3
numcmp,all

allsel,all
plot

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.6 Boundary Conditions and Loads


!!Select apropriate nodes and apply BC
allsel, all
nsel,s,loc,x,-incc,+incc     ! line selection
d,all,ux,0
nsel,s,loc,x,-incc,+incc     ! node selection
nsel,r,loc,y,-incc,+incc
d,all,uy,0
nsel,s,loc,z,-incc,+incc     ! node selection
cp,1,uz,all
nsel,s,loc,z,Eff_P_sb-incc,Eff_P_sb+incc       ! node selection
cp,2,uz,all

nlgeom, on

allsel,all
finish


!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 3. solve

!!  Solve
/solu
!tref,293
t_room            = T_C0
t_low             = T_C1
t_high            = T_C2
time_VAR = 0
time_STEP = 3600

antype,trans,new
nropt,auto,,                              !Specifies the Newton-Raphson options in a static or full transient analysis

! load step 0
tref,t_room
time_VAR = time_VAR + time_STEP*time_RMP
```

```
time, time_VAR
kbc,0                          !specifies stepped or ramped loading within a load step
nsubst,20                      !specifies the number of substeps to be taken this load step.
autots,off                     !specifies whether to use automatic time stepping or load stepping
nropt,full,,on                 !use full Newton-Raphson with adaptive descent
sstif,on                       !include stress stiffening
nlgeom,on                      !include large deformation effects
eqslv,sparse                   !specifies the type of equation solver
toffst,0,                      !set the temp offset from absolute zero to zero
neqit,100                      !set 100 as max number of iterations
allsel
bf,all,temp,t_low              !temperature
outres,all,last                 !put only end of time step results to the database
solve
save

! load step 1-1
time_VAR = time_VAR + time_STEP*time_DLO
time, time_VAR
kbc,0                          !specifies stepped or ramped loading within a load step
nsubst,20                      !specifies the number of substeps to be taken this load step.
autots,off                     !specifies whether to use automatic time stepping or load stepping
nropt,full,,on                 !use full Newton-Raphson with adaptive descent
sstif,on                       !include stress stiffening
nlgeom,on                      !include large deformation effects
eqslv,sparse                   !specifies the type of equation solver
toffst,0,                      !set the temp offset from absolute zero to zero
neqit,100                      !set 100 as max number of iterations
allsel
bf,all,temp,t_low               !temperature
outres,all,last                 !put only end of time step results to the database
solve
save

! load step 1-2
time_VAR = time_VAR + time_STEP*time_RMP
time, time_VAR
kbc,0                          !specifies stepped or ramped loading within a load step
nsubst,20                      !specifies the number of substeps to be taken this load step.
autots,off                     !specifies whether to use automatic time stepping or load stepping
nropt,full,,on                 !use full Newton-Raphson with adaptive descent
sstif,on                       !include stress stiffening
nlgeom,on                      !include large deformation effects
eqslv,sparse                   !specifies the type of equation solver
toffst,0,                      !set the temp offset from absolute zero to zero
neqit,100                      !set 100 as max number of iterations
allsel
bf,all,temp,t_high             !temperature
outres,all,last                !put only end of time step results to the database
solve
save

! load step 1-3
time_VAR = time_VAR + time_STEP*time_DHI
time, time_VAR
kbc,0                          !specifies stepped or ramped loading within a load step
nsubst,20                      !specifies the number of substeps to be taken this load step.
autots,off                     !specifies whether to use automatic time stepping or load stepping
nropt,full,,on                 !use full Newton-Raphson with adaptive descent
sstif,on                       !include stress stiffening
nlgeom,on                      !include large deformation effects
eqslv,sparse                   !specifies the type of equation solver
```

```
toffst,0,                       !set the temp offset from absolute zero to zero
neqit,100                       !set 100 as max number of iterations
allsel
bf,all,temp,t_high              !temperature
outres,all,last                 !put only end of time step results to the database
solve
save

finish

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 4. postprocess

!! Postprocessing
/post1
set,2
allsel
nsel,s,loc,x,Pos_First_sb + Eff_P_sb*(Eff_N_sb-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(Eff_N_sb-1)+Eff_P_sb/2
nsel,r,loc,y,Th_PWB+Th_pad+H_sb-0.025,Th_PWB+Th_pad+H_sb
nsel,r,loc,z,-incc,Eff_P_sb+incc
esln
esel,r,mat,,3
etable,e_x1,eppl,x
etable,e_y1,eppl,y
etable,e_z1,eppl,z
etable,r_xy1,eppl,xy
etable,r_yz1,eppl,yz
etable,r_xz1,eppl,xz
etable,volu1,volu
pretab,e_x1,e_y1,e_z1,e_xy1,e_yz1,e_xz1,volu1

set,last
allsel
nsel,s,loc,x,Pos_First_sb + Eff_P_sb*(Eff_N_sb-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(Eff_N_sb-1)+Eff_P_sb/2
nsel,r,loc,y,Th_PWB+Th_pad+H_sb-0.025,Th_PWB+Th_pad+H_sb
nsel,r,loc,z,-incc,Eff_P_sb+incc
esln
esel,r,mat,,3
etable,e_x2,eppl,x
etable,e_y2,eppl,y
etable,e_z2,eppl,z
etable,r_xy2,eppl,xy
etable,r_yz2,eppl,yz
etable,r_xz2,eppl,xz
etable,volu2,volu
pretab,e_x2,e_y2,e_z2,e_xy2,e_yz2,e_xz2,volu2

sadd,volu,volu1,volu2,1/2,1/2
sadd,de_x,e_x2,e_x1,1,-1
sadd,de_y,e_y2,e_y1,1,-1
sadd,de_z,e_z2,e_z1,1,-1
sadd,dr_xy,r_xy2,r_xy1,1,-1
sadd,dr_yz,r_yz2,r_yz1,1,-1
sadd,dr_xz,r_xz2,r_xz1,1,-1
sadd,diff1,de_x,de_y,1,-1
sadd,diff2,de_y,de_z,1,-1
sadd,diff3,de_x,de_z,1,-1
smult,s_eq1,diff1,diff1
smult,s_eq2,diff2,diff2
```

```
smult,s_eq3,diff3,diff3
smult,s_eq4,dr_xy,dr_xy
smult,s_eq5,dr_yz,dr_yz
smult,s_eq6,dr_xz,dr_xz
sadd,s_eq12,s_eq1,s_eq2,1,1
sadd,s_eq34,s_eq3,s_eq4,1,1.5
sadd,s_eq56,s_eq5,s_eq6,1.5,1.5
sadd,s_eq_a,s_eq12,s_eq34,1,1
sadd,s_eq_b,s_eq_a,s_eq56,1,1
smult,s_eq,s_eq_b,,0.2222,
sexp,eq,s_eq,,0.5,
smult,veq,eq,volu
pretab,s_eq,eq,veq,volu

ssum
*get,tvolu,ssum,,item,volu
*get,tveq,ssum,,item,veq

avg_eq=tveq/tvolu
*CFOPEN,'pred_sb_tm_s1','txt',' '
*VWRITE,avg_eq, , , , , , , , ,
(F10.8)

allsel
nsel,s,loc,x,Pos_First_sb + Eff_P_sb*(Eff_N_sb-1)-Eff_P_sb/2,Pos_First_sb + Eff_P_sb*(Eff_N_sb-
1)+Eff_P_sb/2
!nsel,r,loc,y,Th_PWB-incc,Th_PWB + Th_pad + H_sb + Th_pad+incc
nsel,r,loc,y,Th_PWB+Th_pad-incc,Th_PWB + Th_pad + Th_CBGA
nsel,r,loc,z,-incc,Eff_P_sb+incc
esln
esel,r,mat,,3

plnsol,eppl,eqv,0,1
```

C.5 A Random Vibration-Induced Fatigue Analysis Model for Solder Balls

C.5.1 Modal Analysis to Find Natural Frequency

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!The Modal Analysis to Find the Natural Frequency
!!of BGA
!!by Injoong Kim
!!12/13/05

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 1. startup

finish
/clear,nostart
/title, CBGA Reliability (Vibration)
/RGB,INDEX,100,100,100, 0
/RGB,INDEX, 80, 80, 80,13
/RGB,INDEX, 60, 60, 60,14
/RGB,INDEX, 0, 0, 0,15
/REPLOT
!Units are m,N/m^2 (same as Pa) etc.
!Units

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2. preprocessor

/prep7
!! preference
KEYW,PR_STRUC,1

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.1 define parameters

!eff_sb_number          ! the number of solder balls in a row
!eff_sb_height          ! the solder ball height
!eff_sb_area            ! the area of solder balls
!eff_sb_pitch           ! the distance between two solder balls
!sb_spring_const        ! the spring constant of the solder balls
!sb_density             ! the density of the solder balls

!board_area             ! the board area
!board_Izz              ! the board area moment of inertia
!board_height           ! the board height
!board_length           ! the board length
!board_ex               ! the young's modulus of the board
!board_prxy             ! the Poisson's Ratio of the board
!board_density          ! the density of the board

!package_area           ! the package area
!package_Izz            ! the package area moment of inertia
!package_height         ! the package height
!package_length         ! the package length
!package_ex             ! the young's modulus of the package
!package_prxy           ! the Poisson's Ratio of the package
!package_density        ! the density of the board
```

```
!pos_package                ! the location of the package (the center of the board)

eff_sb_number = 25
eff_sb_height = 8.539999999999999E-4
eff_sb_area = 1.5552847130677973E-5
eff_sb_pitch = 0.00127
sb_spring_const = 5.535648025574797E8
sb_density = 10000.0
board_area = 3.724E-4
board_Izz = 7.1344E-11
board_height = 0.0028
board_length = 0.133
board_ex = 1.9378333333333332E10
board_prxy = 0.136
board_density = 1900.3
package_area = 9.425E-5
package_Izz = 6.605354166666666E-11
package_height = 0.0029
package_length = 0.0325
package_ex = 2.5511E11
package_prxy = 0.3
package_density = 4567.5
pos_package = 0.0665

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.2 define element types

et,1,beam3                  !Board
et,2,COMBIN14,,,2           !Solder Balls
et,3,beam3                  !Package

r,1,board_area,board_Izz,board_height
r,2,sb_spring_const
r,3,package_area,package_Izz,package_height

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.3 create Material models

mp,ex,1,board_ex
mp,prxy,1,board_prxy
mp,dens,1,board_density

mp,dens,2,sb_density

mp,ex,3,package_ex
mp,prxy,3,package_prxy
mp,dens,3,package_density

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.4 build node and elements

!! create nodes and elements corresponding the solder joints
*do,i,1,eff_sb_number,1
n,i,pos_package - (eff_sb_number-1)*eff_sb_pitch/2.0 + (i-1)*eff_sb_pitch,0.0
n,i+eff_sb_number,pos_package - (eff_sb_number-1)*eff_sb_pitch/2.0 + (i-1)*eff_sb_pitch,eff_sb_height
type,2
real,2
mat,2
e,i,i+eff_sb_number
*enddo
```

```
!! create elements corresponding the electronic board
*do,i,1,eff_sb_number-1,1
type,1
real,1
mat,1
e,i,i+1
*enddo

!! create nodes corresponding the electronic board
n,eff_sb_number*2+1,0.0,0.0
n,eff_sb_number*2+2,board_length,0.0

!! create elements corresponding the electronic board
type,1
real,1
mat,1
e,eff_sb_number*2+1,1
e,eff_sb_number,eff_sb_number*2+2

!! create elements corresponding the package
*do,i,1,eff_sb_number-1,1
type,3
real,3
mat,3
e,i+eff_sb_number,i+eff_sb_number+1
*enddo

!! create nodes corresponding the package
n,eff_sb_number*2+3,pos_package - package_length/2,eff_sb_height
n,eff_sb_number*2+4,pos_package + package_length/2,eff_sb_height

!! create elements corresponding the package
type,3
real,3
mat,3
e,eff_sb_number*2+3,eff_sb_number+1
e,eff_sb_number*2,eff_sb_number*2+4

!! define 0 degree of freedom at the board edge
d,eff_sb_number*2+1,all,0
d,eff_sb_number*2+2,all,0

!! reduce the degree of freedom at the package
m,eff_sb_number+1,uy
m,eff_sb_number*2,uy

allsel,all
plot

finish

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 3. solve

!!  Solve
/solu
antype,modal
mxpand,1
modopt,reduc,,,,1
outpr,basic,all
```

```
solve

/post1
set, , ,1, , , ,1,
pldisp,0

*get,freq1,MODE,1,FREQ
*CFOPEN,'pred_sb_rv_s1.frequency','txt',' '
*VWRITE,freq1, , , , , , , , ,
(F10.2)
```

C.5.2 Structural Analysis to Find Solder Ball Stress

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!The Structural Analysis to Find the Solder Balls Stress
!!of BGA
!!by Injoong Kim
!!12/13/05

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 1. startup

finish
/clear,nostart
/title, CBGA Reliability (Vibration)
/RGB,INDEX,100,100,100, 0
/RGB,INDEX, 80, 80, 80,13
/RGB,INDEX, 60, 60, 60,14
/RGB,INDEX, 0, 0, 0,15
/REPLOT
!Units are m,N/m^2 (same as Pa) etc.
!Units

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2. preprocessor

/prep7
!! preference
KEYW,PR_STRUC,1

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.1 define parameters

!eff_sb_number          ! the number of solder balls in a row
!eff_sb_height          ! the solder ball height
!eff_sb_area            ! the area of solder balls
!eff_sb_pitch           ! the distance between two solder balls
!sb_spring_const        ! the spring constant of the solder balls
!sb_density             ! the density of the solder balls

!board_area             ! the board area
!board_Izz              ! the board area moment of inertia
!board_height           ! the board height
!board_length           ! the board length
!board_ex               ! the young's modulus of the board
!board_prxy             ! the Poisson's Ratio of the board
!board_density          ! the density of the board
```

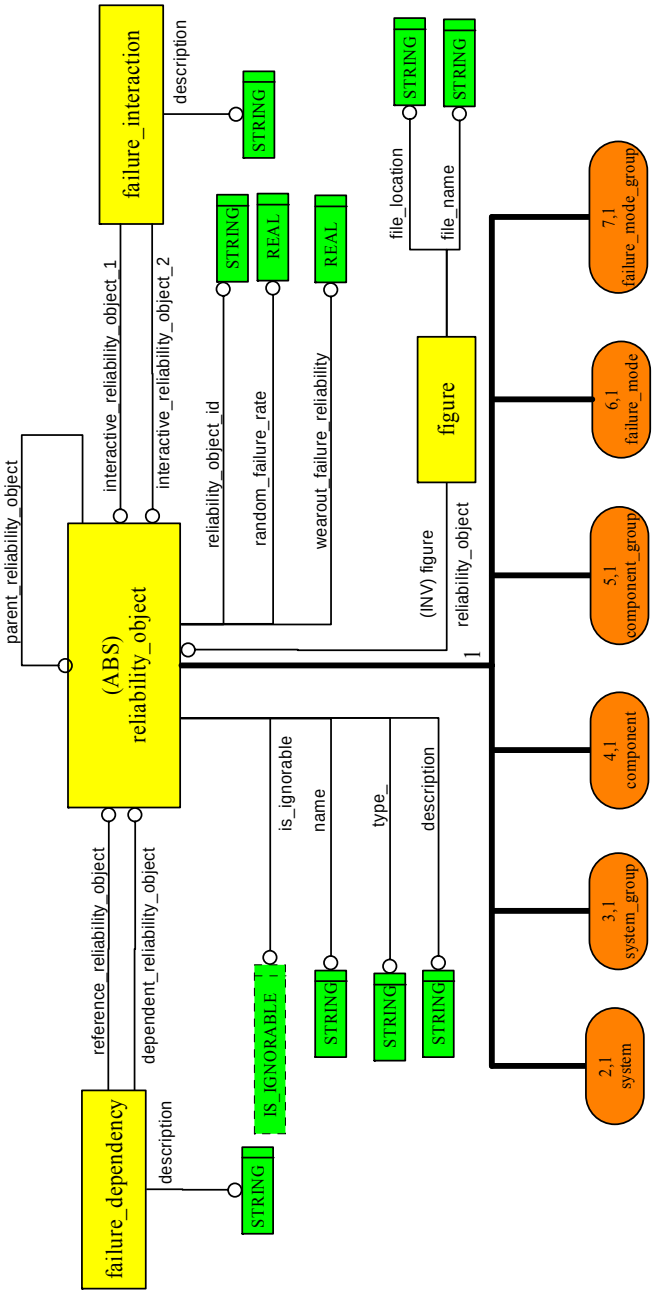
```
!package_area          ! the package area
!package_Izz           ! the package area moment of inertia
!package_height        ! the package height
!package_length        ! the package length
!package_ex            ! the young's modulus of the package
!package_prxy          ! the Poisson's Ratio of the package
!package_density       ! the density of the board

!pos_package           ! the location of the package (the center of the board)
!y_disp                ! the displacement of the board

eff_sb_number = 25
eff_sb_height = 8.539999999999999E-4
eff_sb_area = 1.5552847130677973E-5
eff_sb_pitch = 0.00127
sb_spring_const = 5.535648025574797E8
sb_density = 10000.0
board_area = 3.724E-4
board_Izz = 7.1344E-11
board_height = 0.0028
board_length = 0.133
board_ex = 1.9378333333333332E10
board_prxy = 0.136
board_density = 1900.3
package_area = 9.425E-5
package_Izz = 6.605354166666666E-11
package_height = 0.0029
package_length = 0.0325
package_ex = 2.5511E11
package_prxy = 0.3
package_density = 4567.5
pos_package = 0.0665
y_disp = 6.71693958688946E-5

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.2 define element types

et,1,beam3             !Board
et,2,COMBIN14,,,2      !Solder Balls
et,3,beam3             !Package

r,1,board_area,board_Izz,board_height
r,2,sb_spring_const
r,3,package_area,package_Izz,package_height

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.3 create Material models

mp,ex,1,board_ex
mp,prxy,1,board_prxy
mp,dens,1,board_density

mp,dens,2,sb_density

mp,ex,3,package_ex
mp,prxy,3,package_prxy
mp,dens,3,package_density

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 2.4 build node and elements

!! create nodes and elements corresponding the solder joints
```

```
*do,i,1,eff_sb_number,1
n,i,pos_package - (eff_sb_number-1)*eff_sb_pitch/2.0 + (i-1)*eff_sb_pitch,0.0
n,i+eff_sb_number,pos_package - (eff_sb_number-1)*eff_sb_pitch/2.0 + (i-1)*eff_sb_pitch,eff_sb_height
type,2
real,2
mat,2
e,i,i+eff_sb_number
*enddo

!! create elements corresponding the electronic board
*do,i,1,eff_sb_number-1,1
type,1
real,1
mat,1
e,i,i+1
*enddo

!! create nodes corresponding the electronic board
n,eff_sb_number*2+1,0.0,0.0
n,eff_sb_number*2+2,board_length,0.0

!! create elements corresponding the electronic board
type,1
real,1
mat,1
e,eff_sb_number*2+1,1
e,eff_sb_number,eff_sb_number*2+2

!! create elements corresponding the package
*do,i,1,eff_sb_number-1,1
type,3
real,3
mat,3
e,i+eff_sb_number,i+eff_sb_number+1
*enddo

!! create nodes corresponding the package
n,eff_sb_number*2+3,pos_package - package_length/2,eff_sb_height
n,eff_sb_number*2+4,pos_package + package_length/2,eff_sb_height

!! create elements corresponding the package
type,3
real,3
mat,3
e,eff_sb_number*2+3,eff_sb_number+1
e,eff_sb_number*2,eff_sb_number*2+4

!! define 0 degree of freedom at the board edge
d,eff_sb_number*2+1,all,0
d,eff_sb_number*2+2,all,0

!! define displacement at the center of the board
d,(eff_sb_number+1)/2,uy, y_disp

!! define only y-directional degree of freedom on the package
*do,i,1,eff_sb_number,1
d,eff_sb_number+i,ux,0
*enddo

allsel,all
plot
```

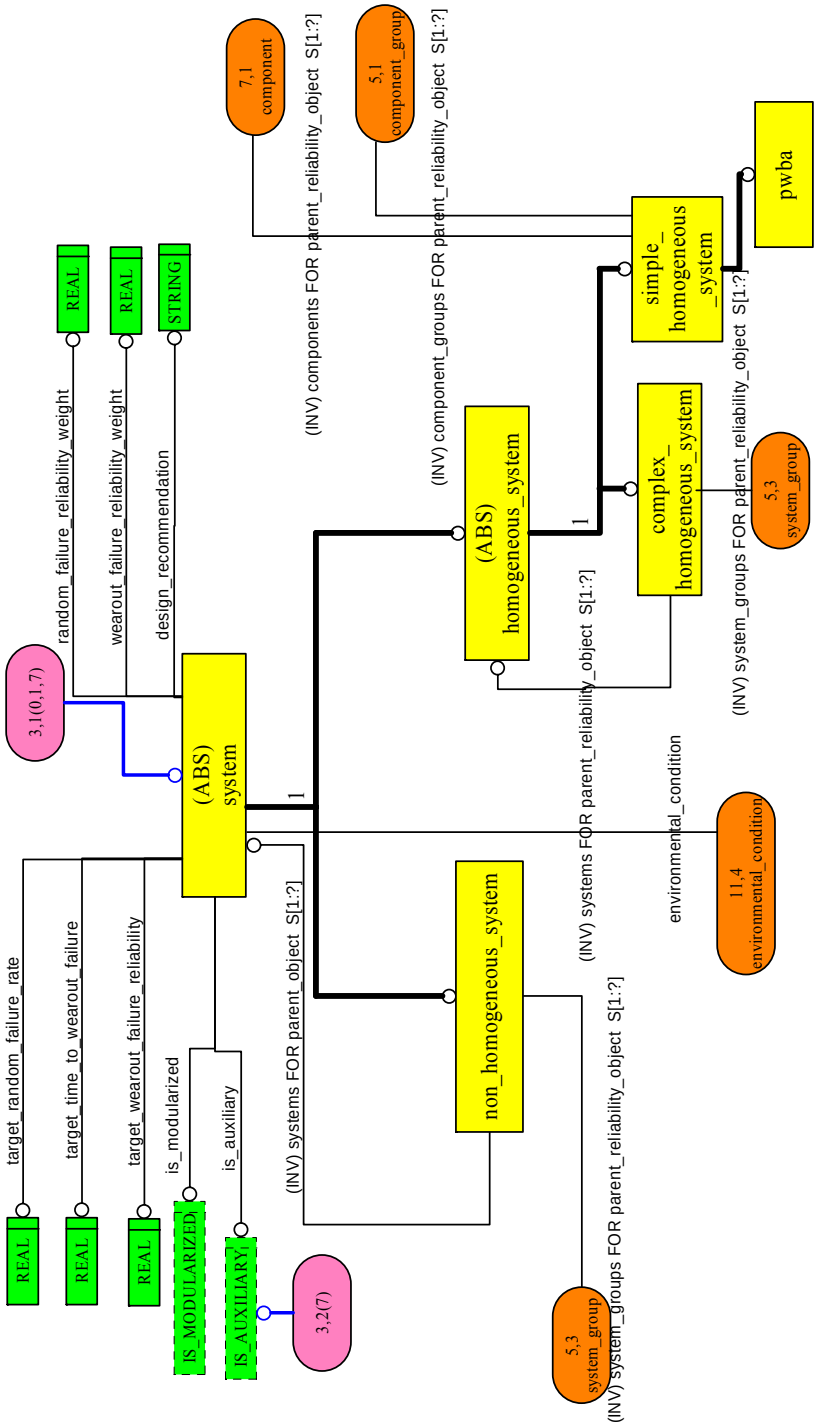
```
finish

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! 3. solve

!!  Solve
/solu
antype,static

solve
finish

/post1
pldisp,2

esel,s,type,,2
etable,sb_f,smisc,1
smult,sb_s,sb_f,,1/eff_sb_area,

esort,etab,sb_s,0,1,,
*GET,maxStr,sort,,MAX
*CFOPEN,'pred_sb_rv_s1.stress','txt',' '
*VWRITE,maxStr, , , , , , , , ,
(F20.2)
```

APPENDIX D

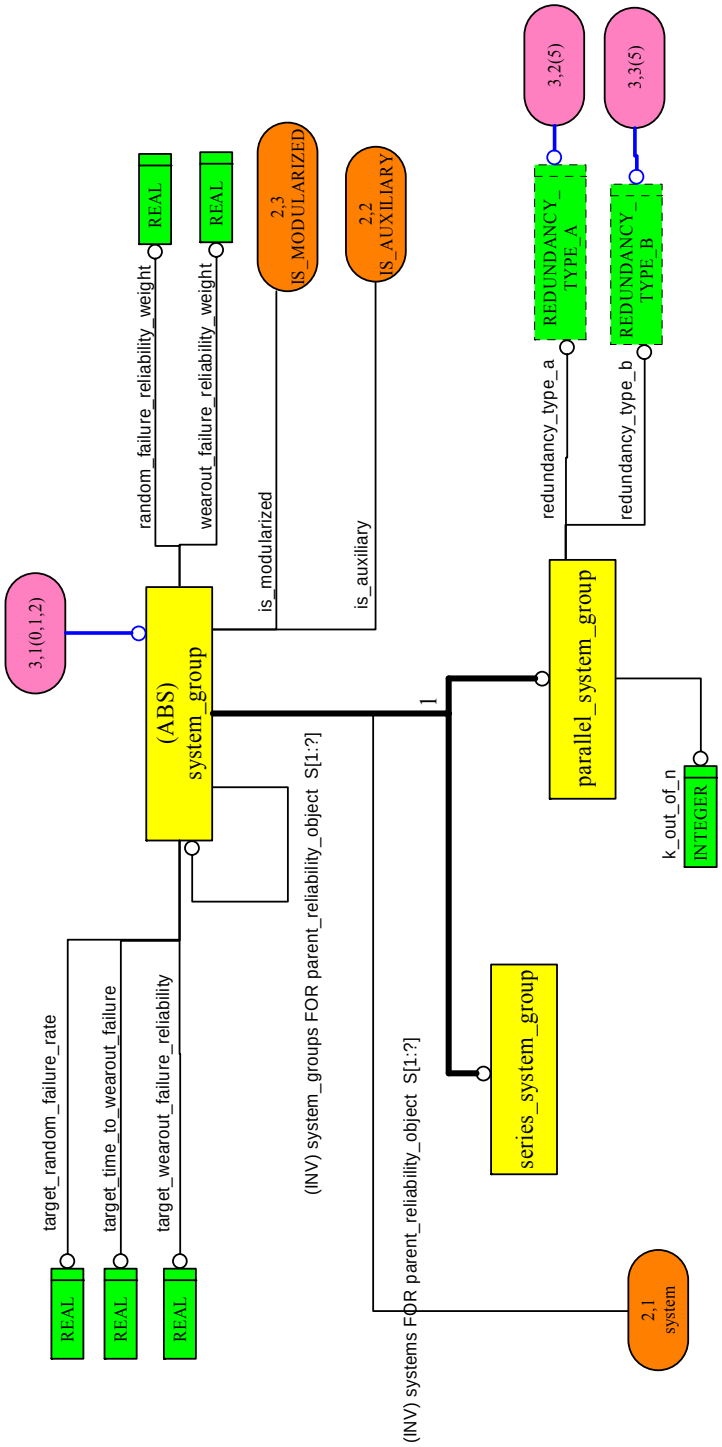
EXPRESS-G DIAGRAMS FOR SDFR-PWBA



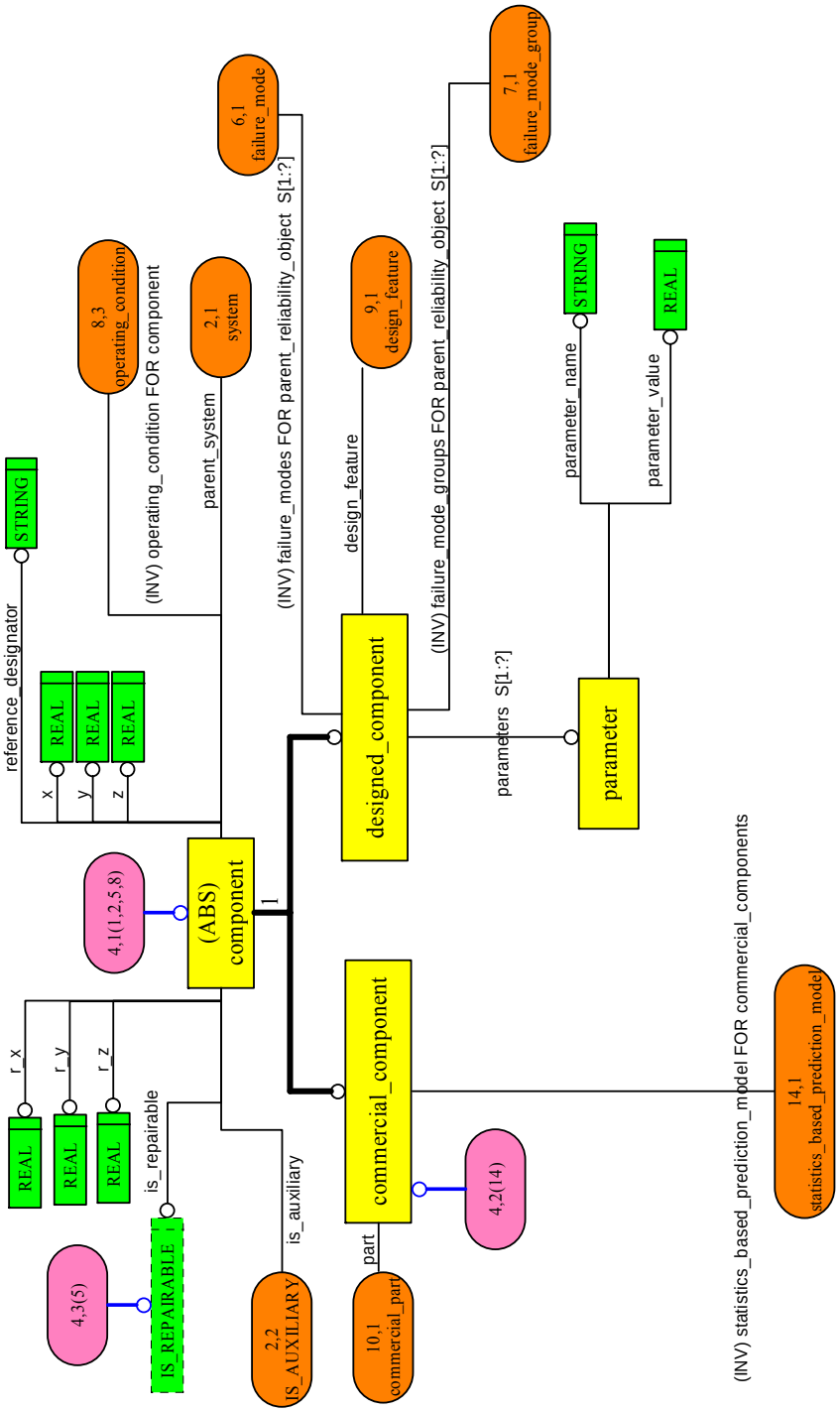
EXPRESS-G Diagram (Page 1 out of 15)



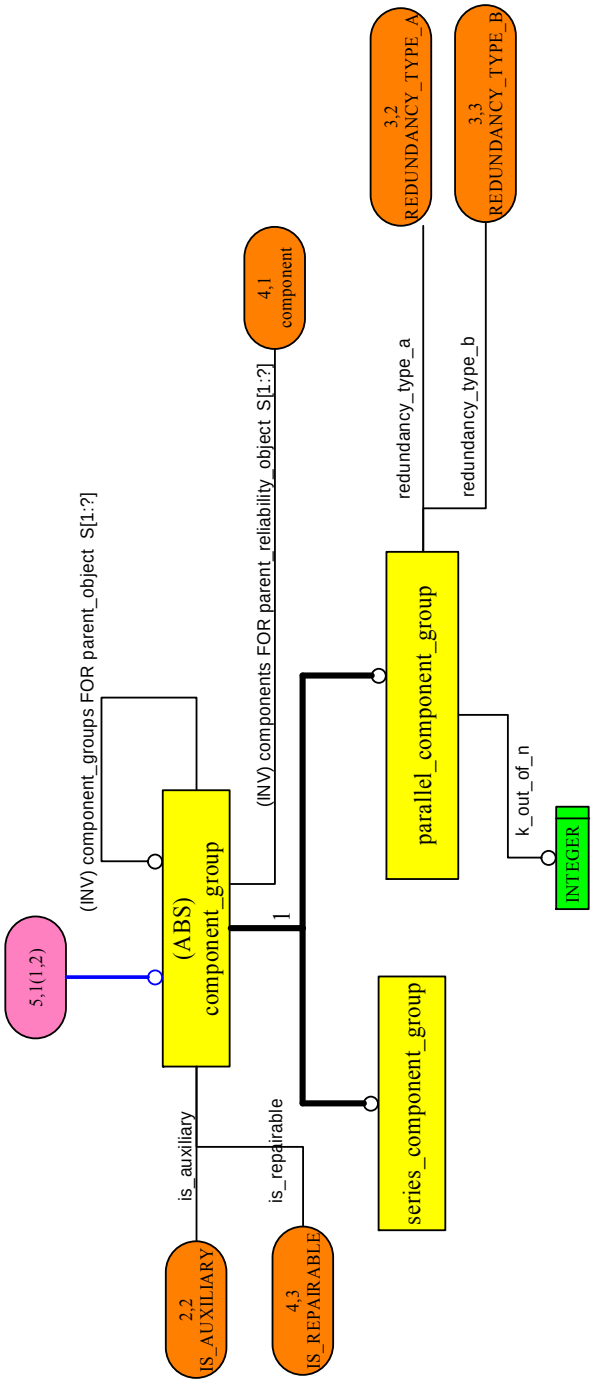
EXPRESS-G Diagram (Page 2 out of 15)



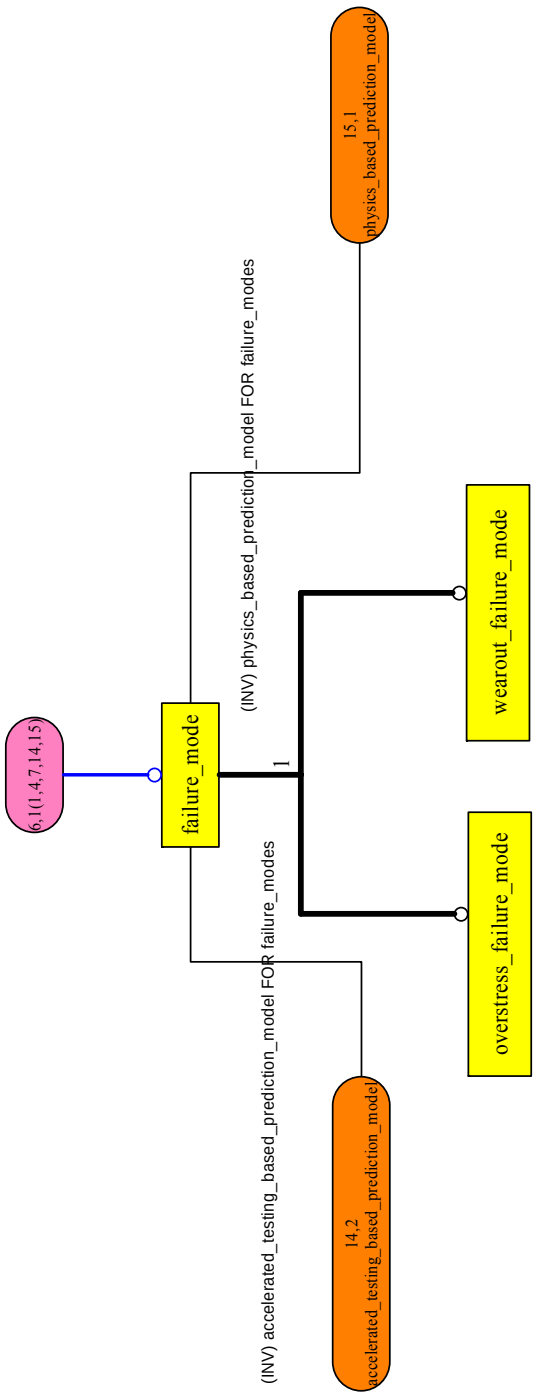
EXPRESS-G Diagram (Page 3 out of 15)



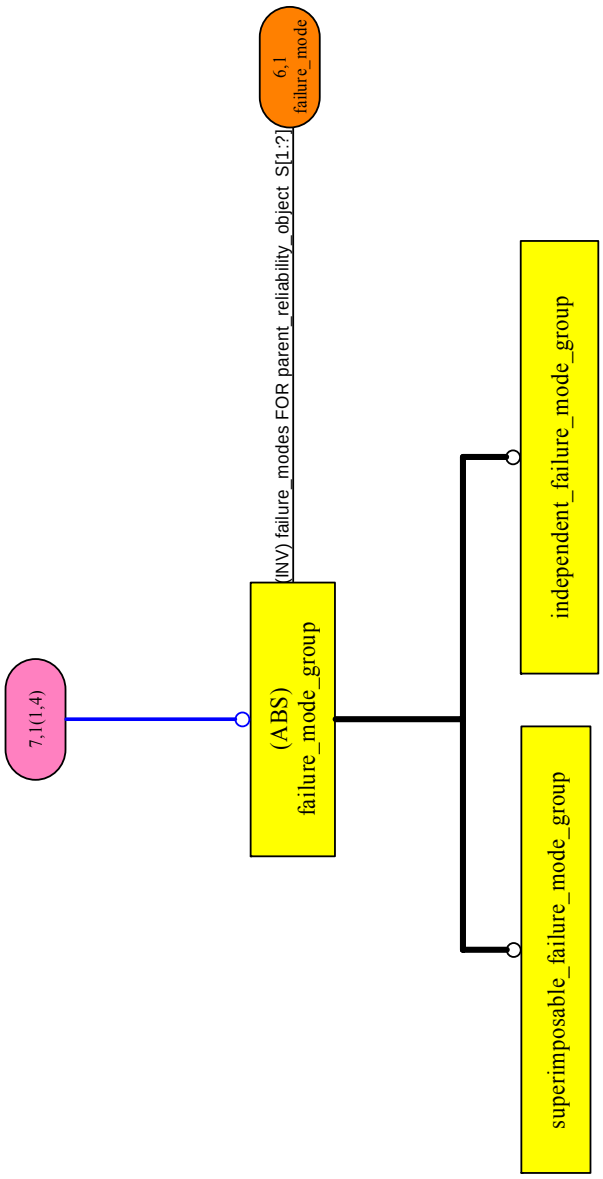
EXPRESS-G Diagram (Page 4 out of 15)



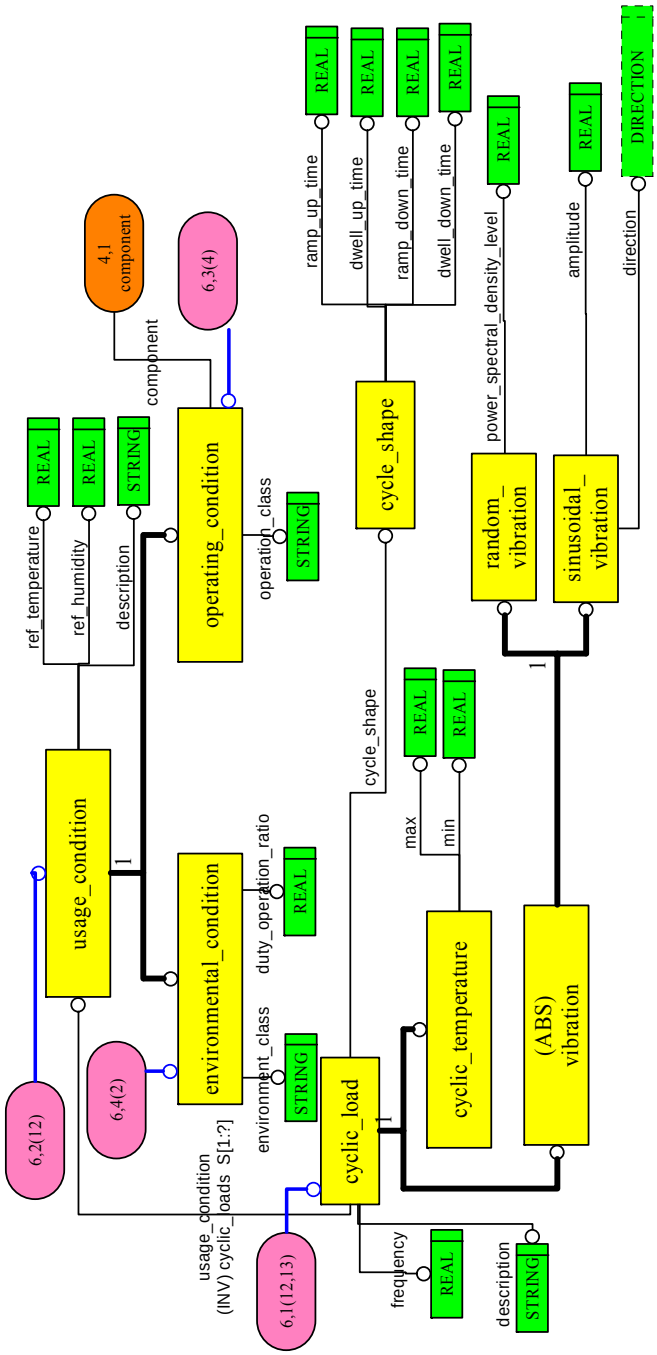
EXPRESS-G Diagram (Page 5 out of 15)



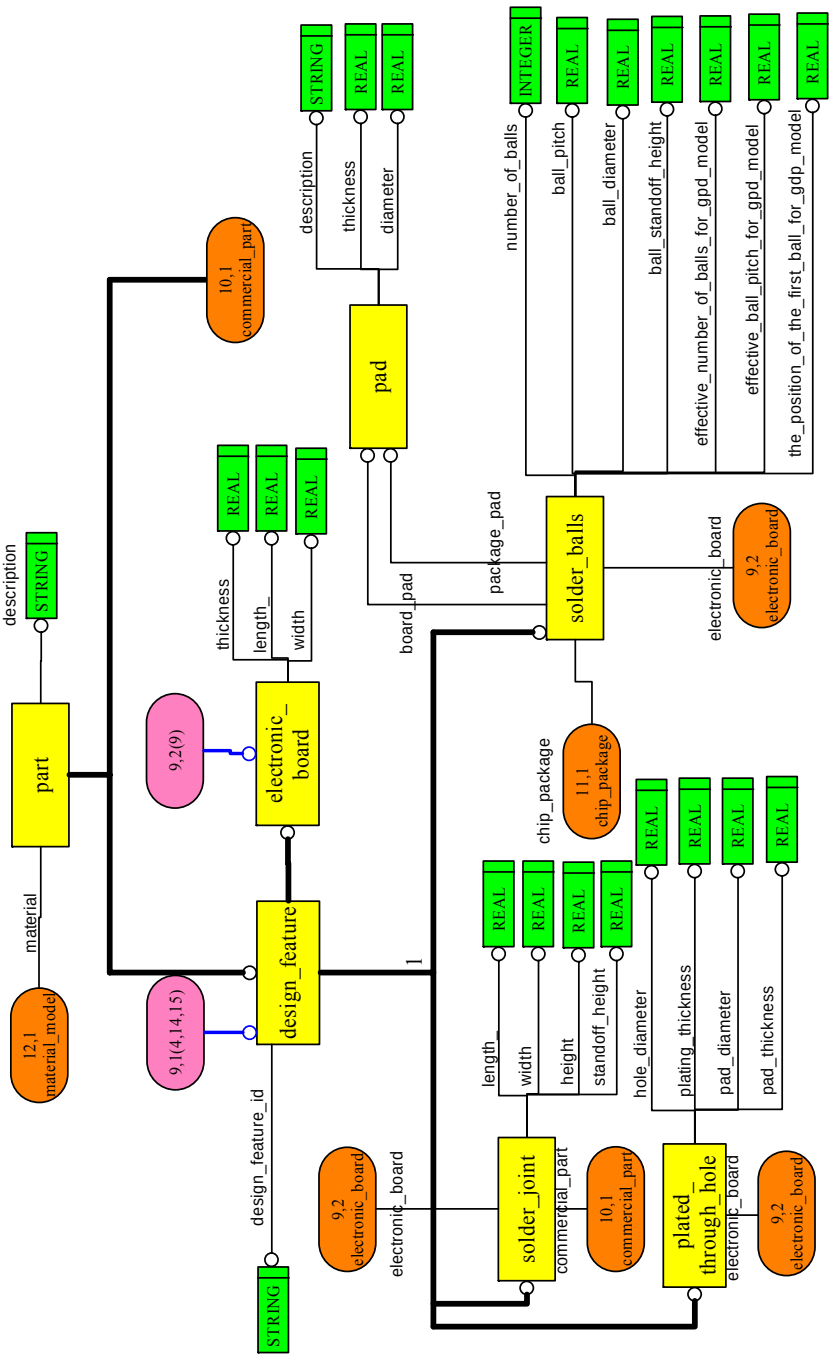
EXPRESS-G Diagram (Page 6 out of 15)



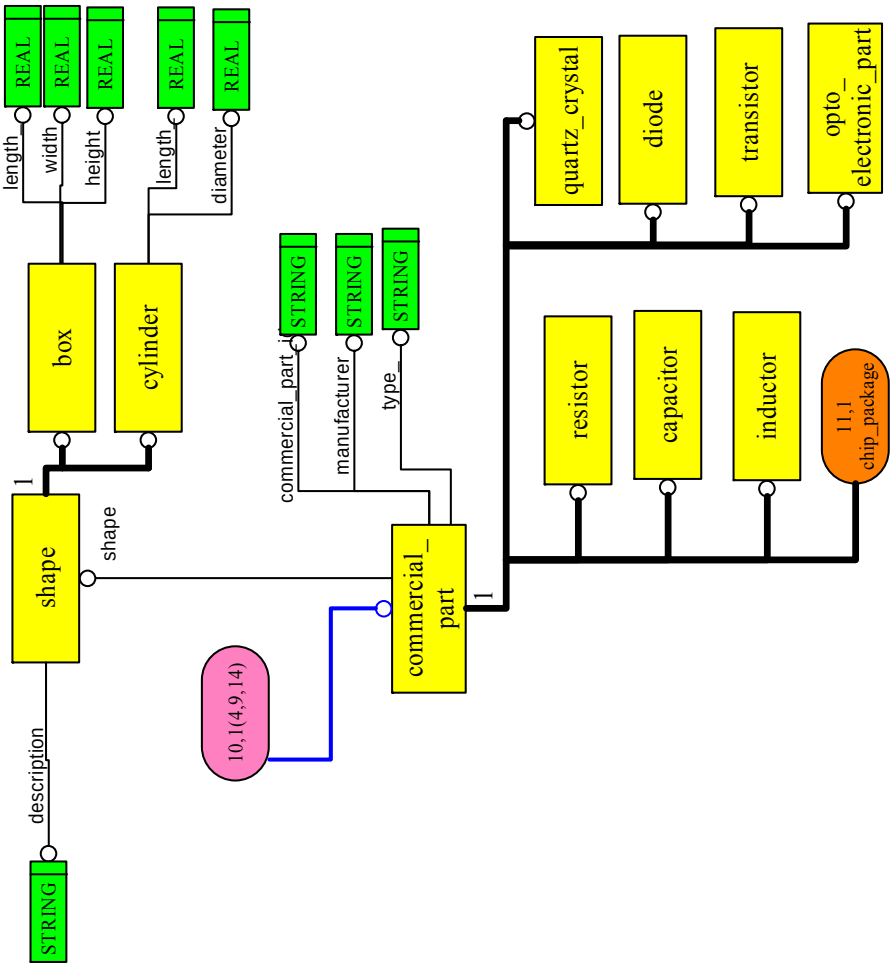
EXPRESS-G Diagram (Page 7 out of 15)



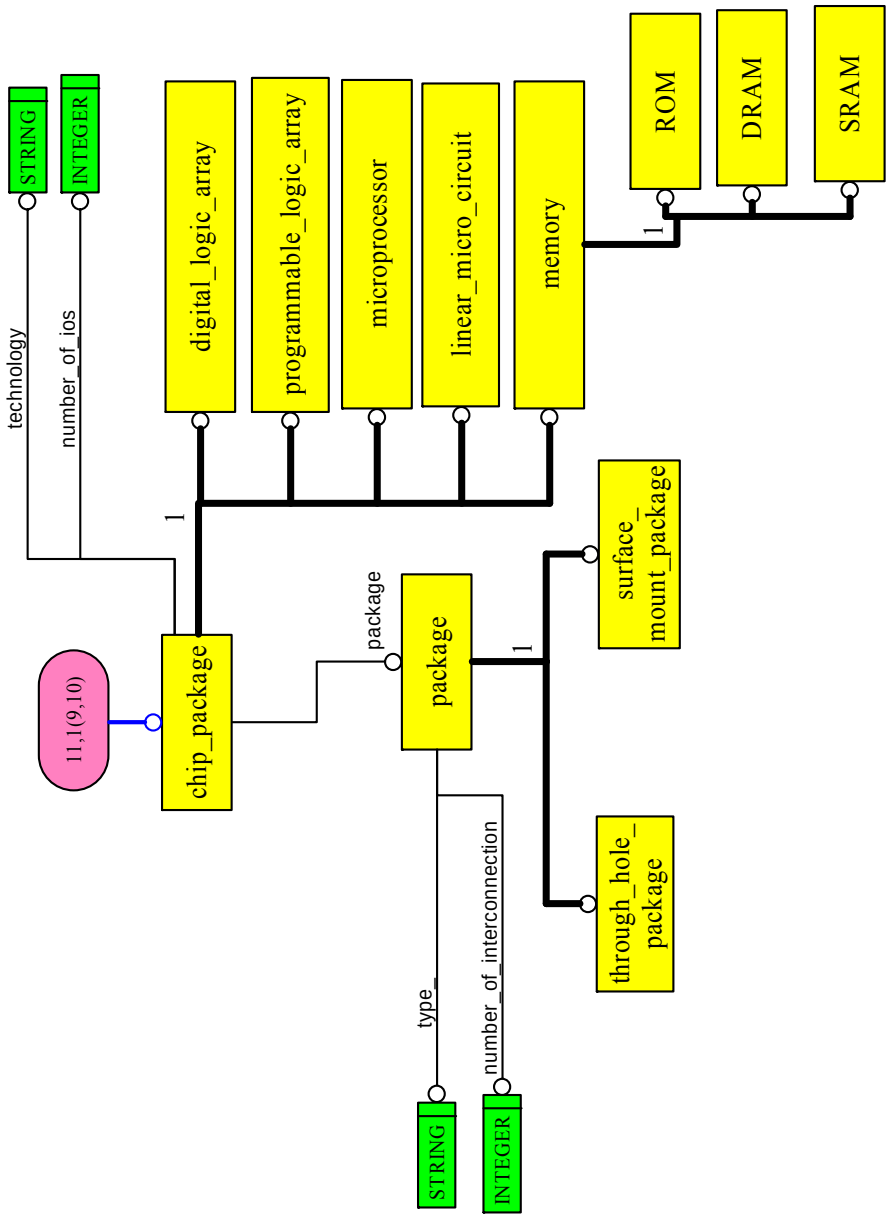
EXPRESS-G Diagram (Page 8 out of 15)



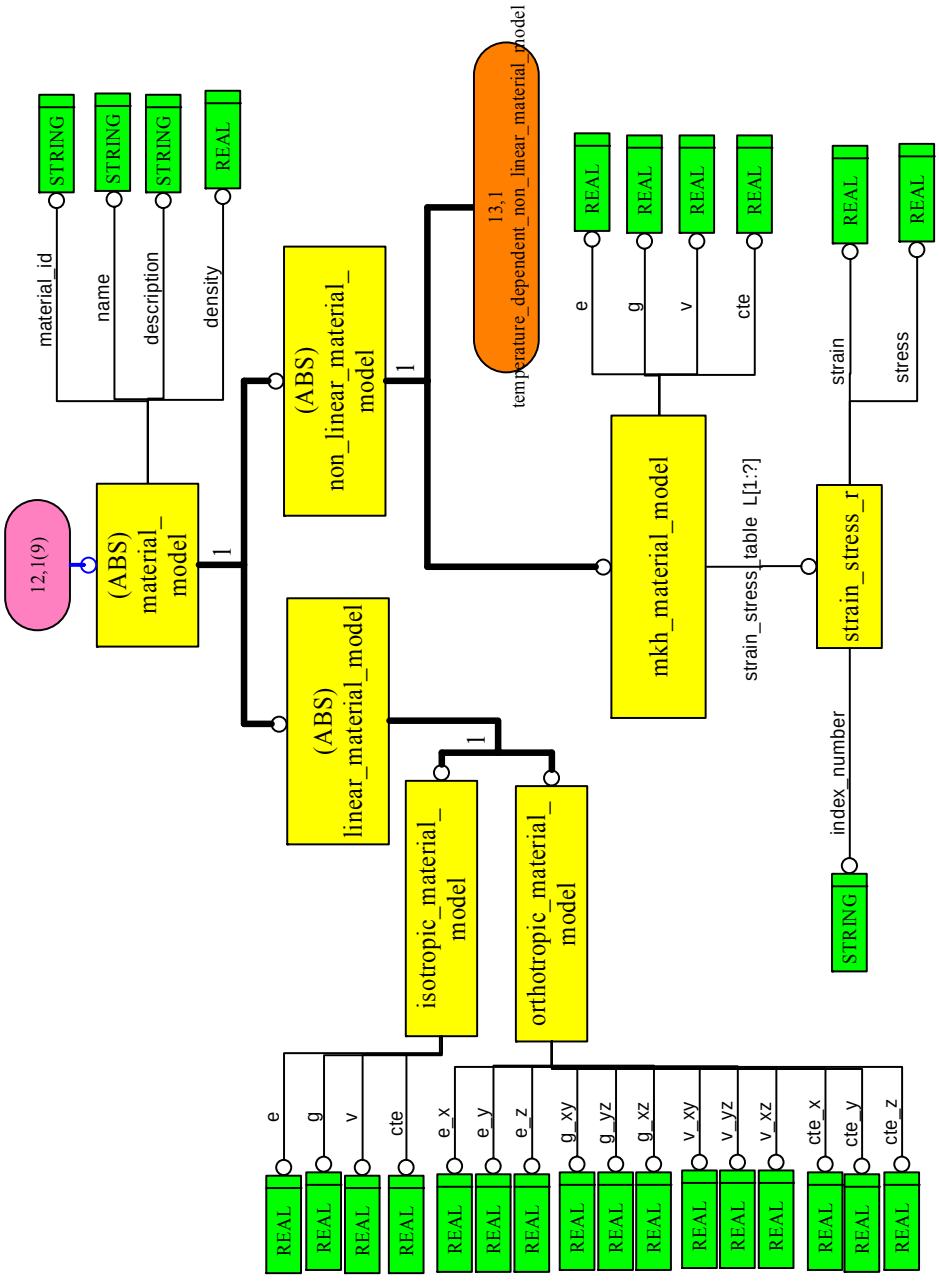
EXPRESS-G Diagram (Page 9 out of 15)



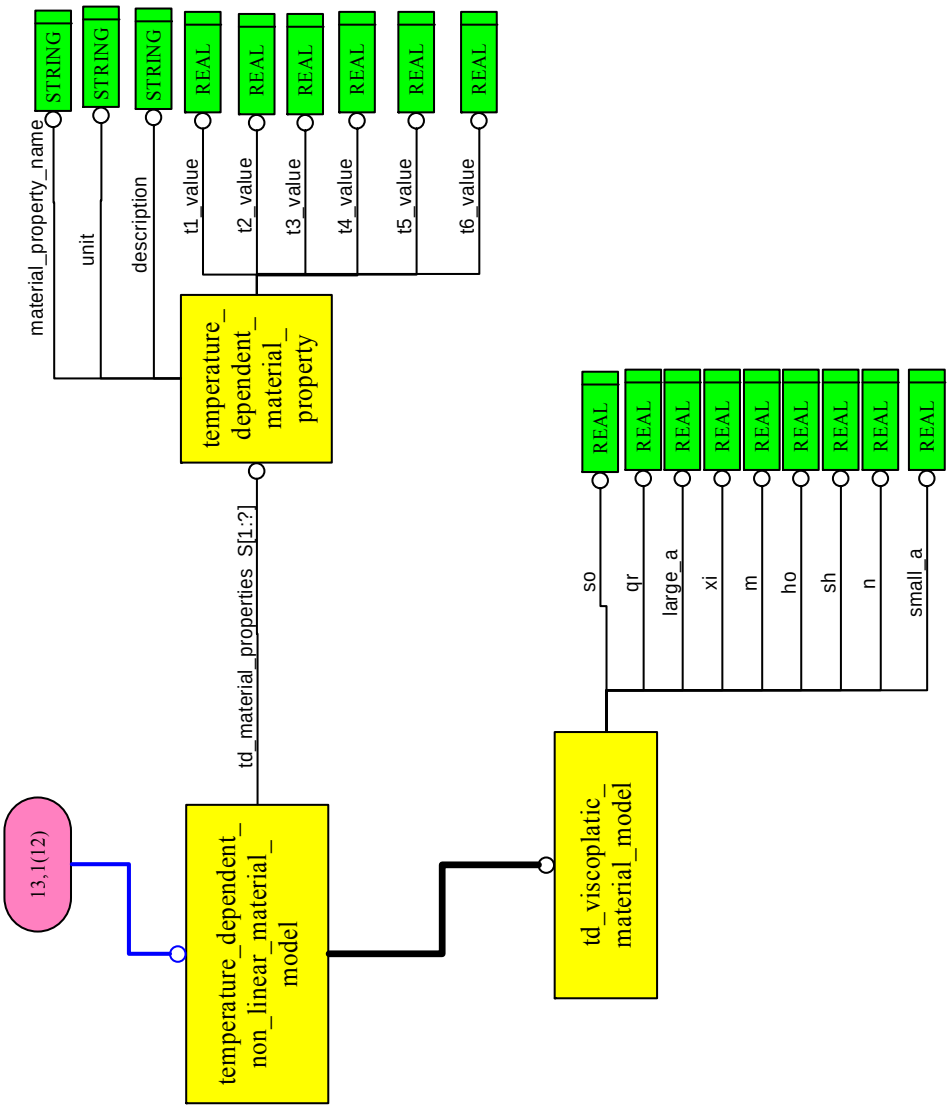
EXPRESS-G Diagram (Page 10 out of 15)



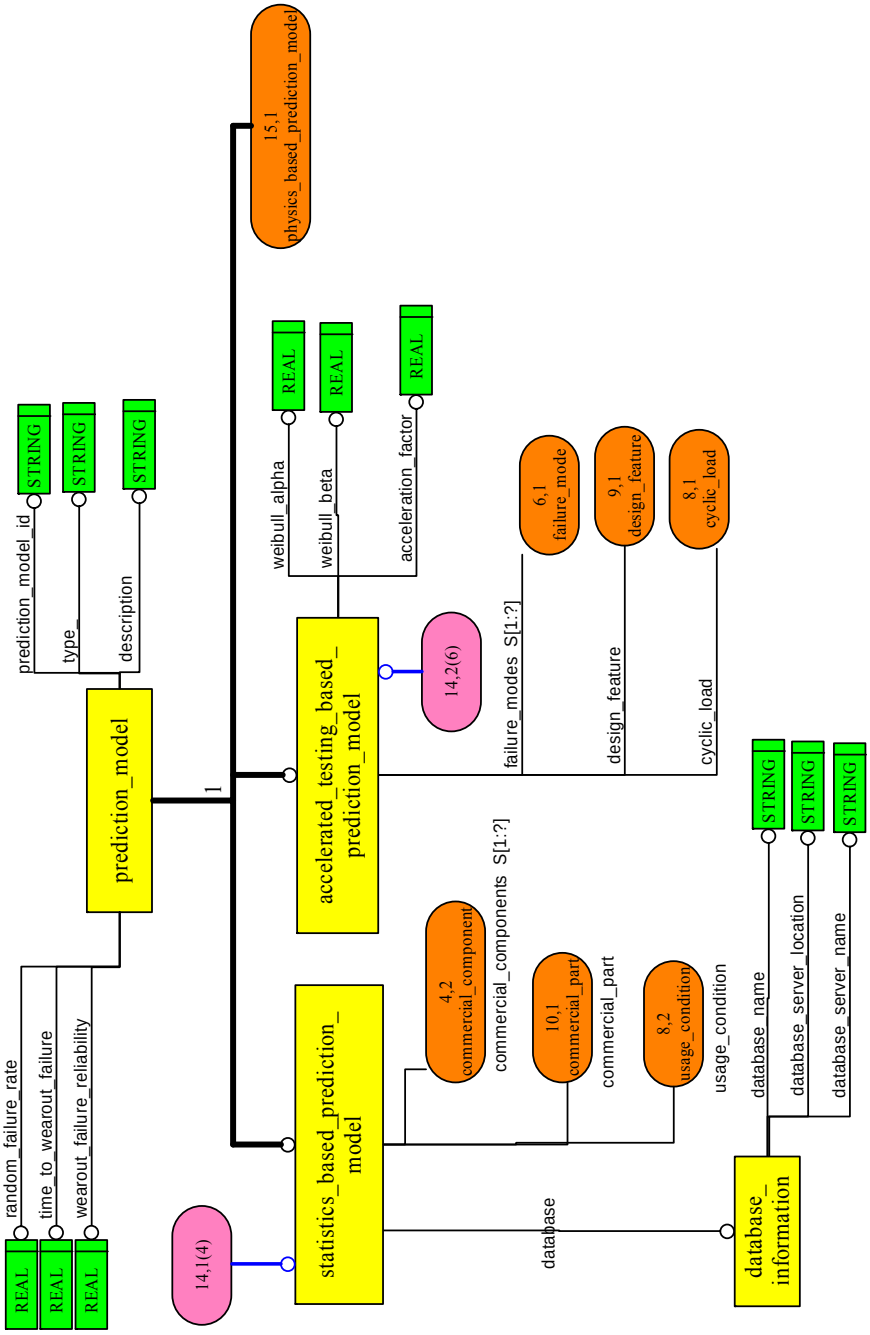
EXPRESS-G Diagram (Page 11 out of 15)



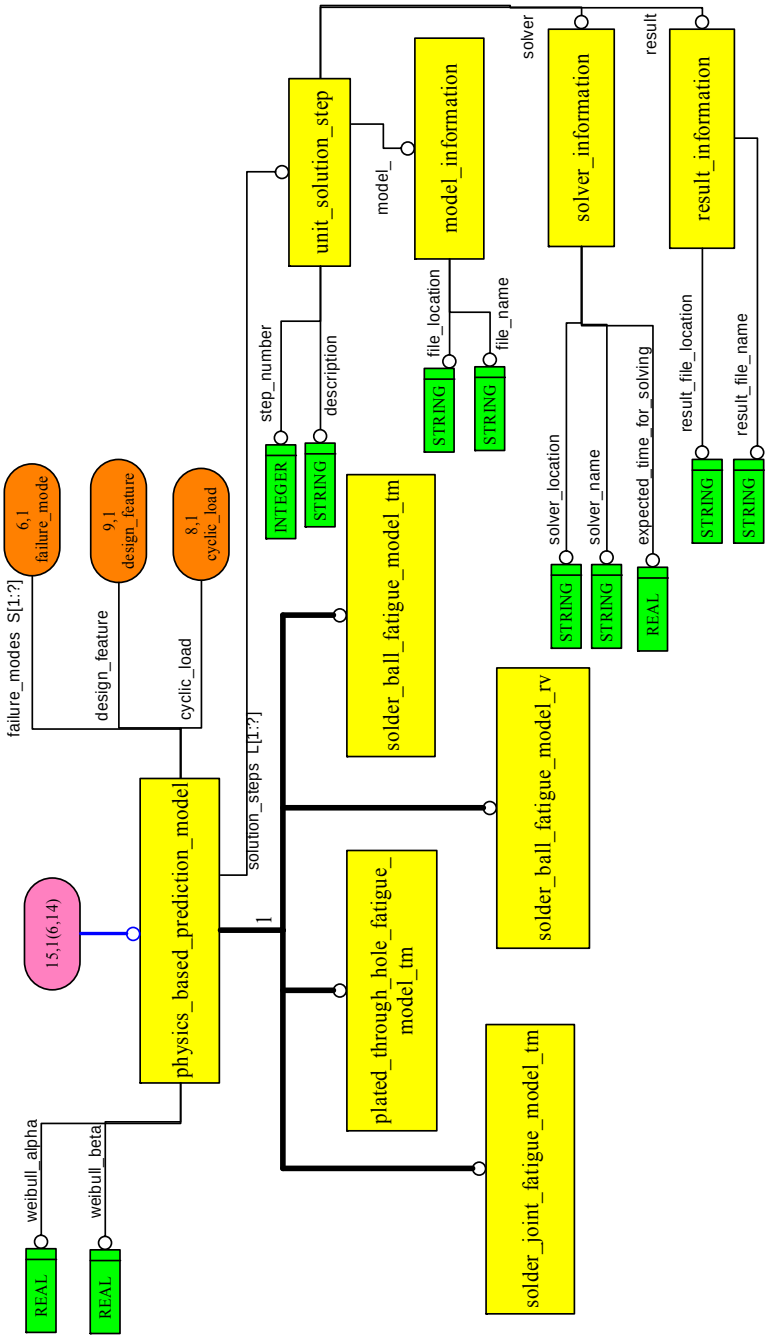
EXPRESS-G Diagram (Page 12 out of 15)



EXPRESS-G Diagram (Page 13 out of 15)



EXPRESS-G Diagram (Page 14 out of 15)



EXPRESS-G Diagram (Page 15 out of 15)

REFERENCES

Ahmad, J., Sitaraman, S. K., 2002, "Modeling Methodologies to Study PWB Assembly Reliability," *52nd Electronic Components and Technology Conference,* San Diego, CA, 1658-1664.

Amari, S. [Online],2006, Available: http://www.relex.com/resources/art/art_datalinking.asp. (date accessed: 4/2007)

ANSYS [Online],2005, Available: http://www.ansys.com. (date accessed: 4/2007)

Baader, F., 2003, "The Description Logic Handbook: Theory, Implementation, and Applications", Cambridge University Press, Cambridge, UK.

Bajaj, M., Peak, R., Wilson, M., et al., 2003, "Towards, Next-Generation Design-for-Manufacturability (DFM) Frameworks for Electronics Product Realization," IEMT of IEEE/CPMT/SEMI.

Bajenescu, T. I., Bazu, M. I., 1999, "Reliability of Electronic Components", Springer-Verlag Berlin Heidelberg, New York.

Bestory, C., Marc, F., Levi, H., 2007, "Statistical Analysis during the Reliability Simulation," Microelectonics Reliability 47(9-11), 1353-1357.

Birolini, A., 2004, "Reliability Engineering: Theory and Practice", Springer, New York.

Blattau, N., Hillman, C., 2005, "A Comparison of the Isothermal Fatigue Behavior of SN-AU-CU to SN-PB Solder," *SMTAI Conference*.

Blischke, W. R., Murthy, D. N. P., 2000, "Reliability: Modeling, Prediction and Optimization", John Wiley & Sons, Inc., New York.

Booker, J. M., McNamara, L. A., 2005, "Expert Knowledge in Reliability Characterization: A Rigorous Approach to Eliciting, Documenting, and Analyzing Expert Knowledge," in *Engineering Design Reliability HandBook*, E. Nikolaidis, D. M. Ghiocel, and S. Singhal, Eds. New York: CRC Press, pp. 31.1-13.13.

Buede, D. M., 2000, "The Engineering Design of Systems", John Wiley & Sons, New York.

Carter, S. [Online],1996, Available: http://www.taygeta.com/rwalks/node7.html. (date accessed: 4/2007)

Chen, P., 1976, "The Entity-Relationship Model - Toward a Unified View of Data," ACM Transactions on Database Systems 1(1), 9-36.

Chenault, R. [Online],2006, Available: http://www.helis.com/stories/ah64afgh.php. (date accessed: 4/2007)

Cho, J., Han, S., Kim, H., 2006, "Meta-Ontology for Automated Information Integration of Parts Libraries," Computer-Aided Design 38, 713-725.

Condra, L. W., 1993, "Reliability Improvement with Design of Experiments", Marcel Dekker, New York.

Covington, M. A., Nute, D., Vellino, A., 1996, "Prolog Programming in Depth", Prentice Hall, New York.

Crow, K. [Online],2002, Available: http://www.npd-solutions.com/fmea.html. (date accessed: 4/2007)

Crowe, D., Feinberg, A., 2001, "Design for Reliability", CRC Press, New York.

Denson, W., 1998, "The History of Reliability Prediction," *IEEE Transactions on Reliability* 47(3), 321-328.

Dhingra, A. K., 1992, "Optimal Apportionment of Reliability & Redundancy in Series Systems under Multiple Objectives," *IEEE Transactions on Reliability* 41(4), 576-582.

DoD, 1989, "Environmental Test Methods and Engineering Guidelines (MIL-STD-810E)," Department of Defense, Wright-paterson AFB.

DoD, 1991, "Reliability Prediction of Electronic Equipment (MIL-HDBK-217F)," Department of Defense, Washington DC.

Dowling, N. E., 1999, "Mechanical Behavior of Materials", Prentice-Hall, Upper Saddle River.

Eastman, C. M., Fereshetian, N., 1994, "Information Models for Use in Product Design: a Comparison," Computer-Aided Design 26, 551-572.

Ebel, G. H., 1998, "Reliability Physics in Electronics: A Historical View," *IEEE Transactions on Reliability* 47(3), 379-389.

Elegbede, A. O. C., Chu, C., Adjallah, K. H., Yalaoui, F., 2003, "Reliability Allocation through Cost Minimization," *IEEE Transactions on Reliability* 52(1), 106-110.

Elmasri, R., Navathe, S. B., 1994, "Fundamentals of Database Systems", Addison-Wesley Publishing Company, Melno Park.

Engelmaier, W., 1983, "Fatigue Life of Leadless Chip Carrier Solder Joints during Power Cycling," *IEEE Transaction on Components, Hybrids, and Manufacturing Technology* CHMT-6(3), 232-237.

Engelmaier, W., 1987, "Results of the IPC Copper Foil Ductility Round-Robin Study," IPC Publication No. 947.

Evans, J., Lall, P., Bauernschub, R., 1995, "A Framework for Reliability Modeling Electronics," *Proceedings of the 1995 Annual Reliability and Maintainability Symposium*, Washington, DC, USA, 144-151.

Falcone, D., Silvestri, A., Bona, G. d., 2002, "Integrated Factors Method for Reliability Allocation: Application to an Aerospace Prototype Project," *Software Engineering and Application*, ACTA Press, 374-166.

Fenves, S. J., Choi, Y., Gurumoorthy, B., Mocko, G., Sriram, R. D., 2003, "Master Product Model for the Support of Tighter Integration of Spatial and Functional Design," NISTIR 7004, NIST.

Foucher, B., Boullie, J., Meslet, B., Das, D., 2002, "A Review of Reliability Prediction Methods for Electronic Devices," *Microelectronics Reliability* 42, 1155-1162.

Fried, I. [Online],2000, Available: http://news.com.com/2100-1001-245029.html. (date accessed: 4/2007)

Geymayr, J. A. B., Ebecken, N. F. F., 1995, "Fault-Tree Analysis: A Knowledge-Engineering Approach," *IEEE Transactions on Reliability* 44(1), 37-45.

Giarratano, J., Riley, G., 1994, "Expert Systems", International Thomson Publishing, Boston.

Gorti, S. R., Gupta, A., Kim, G. J., Sriram, R. D., Wong, A., 1998, "An Object-Oriented Representation for Product and Design Processes," *Computer-Aided Design* 30(7), 489-501.

Grosse, I. R., Milton-Benoit, J. M., Wileden, J. C., 2005, "Ontologies for Supporting Engineering Analysis Methods," Artificial Intelligence for Engineering Design, Analysis and Manufacturing 19(1), 1-18.

Harmon, P., Watson, M., 1997, "Understanding UML: The Developer's Guide", Morgan Kaufmann Publishers, Inc., San Francisco.

Healy, J. D., Jain, A. K., Bennett, J. M., 1997, "Reliability Prediction," *Reliability and Maintainability Symposium*.

IBM [Online],2006, Available: http://www.ibm.com. (date accessed: 4/2007)

IEEE [Online],1990, Available: http://www.tricare.osd.mil/jmis/download/EA-Ref/IEEE61012-1990.pdf. (date accessed: 4/2007)

IPC-TP-510, 1984, "Thermal Induced Strain in Plated Through Holes," Institute for Interconnecting and Packaging Electronic Circuits, Evanston, IL.

ISO [Online],2007, Available: http://www.iso.org/iso/en/ISOOnline.frontpage. (date accessed: 4/2007)

Item [Online],2006, Available: http://www.itemuk.com. (date accessed: 4/2007)

Janasak, K. M., 2001, "Practical Solution to Enterprise Reliability CAE Problems," *Reliability and Maintainability Symposium*, 75-79.

Jensen, F., 1995, "Electronic Component Reliability", John Wiley & Sons, New York.

Jones, J., Marshall, J., Aulak, G., Newman, B., 2003, "Development of An Expert System For Reliability Task Planning As Part Of The REMM Methodology," *Reliability and Maintainability Symposium*, 423-428.

Kapur, K. C., Lamberson, L. R., 1977, "Reliability in Engineering Design", Wiley, New York.

Kartik, S., Murthy, C. S. R., 1995, "Improved Task-Allocation Algorithms to Maximize Reliability of Redundant Distributed computing Systems," *IEEE Transaction on Reliability* 44(4), 575-586.

Kemmerer, S. J., Ed. 1999, "STEP: The Grand Experience." Washington, NIST Special Publication 939.

Kim, I., Sitaraman, S. K., Peak, R. S., 2005, "Reliability Object Model: A Knowledge Model of System Design for Reliability," *ASME International Mechanical Engineering Congress and Exposition IMECE2005-79934*.

Kim, I., Pucha, R. V., Peak, R. S., Sitaraman, S. K., 2007, "System-Design-for-Reliability Tools for Highly Integrated Electronic Packaging Systems," Electronic Components and Technology Conference, 1809-1814.

Kuo, W., Prasad, V. R., 2000, "An Annotated Overview of System-Reliability Optimization," *IEEE Transaction on Reliability* 49(2), 176-187.

Lau, J. H., Rice, D. W., Avery, P. A., 1986, "Nonlinear Analysis of Surface Mount Solder Joint Fatigue," *IEEE CHMT Symposium*, 173-184.

Leemis, L. M., 1995, "Reliability: Probability Models and Statistical Methods", Prentice-Hall, Upper Saddle River.

LKSoft [Online],2006, Available: http://www.lksoft.com. (date accessed: 4/2007)

Mathworks [Online],2005, Available: http://www.mathworks.com. (date accessed: 4/2007)

Mercado-Corujo, H.,2001,"Study of the Thermo-Mechanical Reliability of a Plated-Through-Hole / Press-Pin Assembly," *Master Thesis in Mechanical Engineering. Atlanta: Georgia Institute of Technology.*

Mettas, A., 2000, "Reliability Allocation and Optimization for Complex Systems," *Reliability and Maintainability Symposium*, 216-221.

Mocko, M. G.,2006,"A knowledge Framework for Integrating Multiple Perspective in Decision-Centric Design," *Ph.D. Thesis in Mechanical Engineering. Atlanta: Georgia Institute of Technology*.

NORGE [Online],2007, Available: http://www.standard.no. (date accessed: 4/2007)

NTSB [Online],2006, Available: http://www.ntsb.gov. (date accessed: 4/2007)

O'Connor, P. D. T. [Online],2007, Available: http://www.pat-oconnor.co.uk/reltestst ds.htm. (date accessed: 4/2007)

Oracle [Online],2006, Available: http://www.oracle.com. (date accessed: 4/2007)

Osterman, M., Stadterman, T., 1999, "Failure Assessment Software for Circuit Card Assemblies," *Reliability and Maintainability Symposium*, 269-276.

Painton, L., Campbell, J., 1995, "Genetic Algorithms in Optimization of System Reliability," *IEEE Transactions on Reliability* 44(2), 172-178.

Park, K. S., 1987, "Fuzzy apportionment of system reliability," *IEEE Transactions on Reliability* 36, 129-132.

Patterson-Hine, F. A., Koen, B. V., 1989, "Direct Evaluation of Fault Trees Using Object-Oriented Programming Techniques," *IEEE Transactions on Reliability* 38, 186-192.

Peak, R. S., Fulton, R. E., Nishigaki, I., Okamoto, N., 1998, "Integrating Engineering Design and Analysis Using a Multi-Representation Approach," *Engineering with Computers* 14(2), 93-114.

Peak, R. S., Lubell, J., Srinivasan, V., Waterbury, S. C., 2004, "STEP, XML, and UML: Complementary Technologies," *Journal of Computing & Information Science in Engineering* 4(4), 379-390.

Penoyer, J. A., Burnett, G., Fawcett, D. J., Liou, S.-Y., 2000, "Knowledge Based Product Life Cycle Systems: Principles of Integration of KBE and C3P," *Computer-Aided Design* 32(5-6), 311-320.

Perkins, A., Sitaraman, S. K., 2003, "Thermo-Mechanical Failure Comparison and Evaluation of CCGA and CBGA Electronic Packages," *53rd Electronic Components and Technology Conference*, New Orleans LA, United States, 422-430.

Perkins, A., Sitaraman, S. K., 2004, "Vibration-Induced Solder Joint Fatigue of a Ceramic Column Grid Array (CCGA) Package," *54th Electronic Components and Technology Conference*, Las Vegas, NV, United States, 1271-1278.

Pratt, M. J., Anderson, B. D., Ranger, T., 2005, "Towards the Standardized Exchange of Parameterized Feature-Based CAD Models," Computer-Aided Design 37, 1251-1265.

Pucha, R. V., Hedge, S., Damani, M., et al., 2004, "System-Level Reliability Assessment of Mixed-Signal Convergent Microsystems," *IEEE Transactions on Advanced Packaging* 27(2), 438-452.

Raghunathan, R.,2000,"Virtual Qualification Methodology for Next-Generation Area-Array Packages," *Master Thesis in Mechanical Engineering. Atlanta: Georgia Institute of Technology*.

Rao, S. S., 1992, "Reliability-Based Design", McGraw-Hill, Inc., New York.

Rassaian, M., Lee, J., 2004, "Generalized Multi-Domain Method for Fatigue Analysis of Interconnect Structures," Finite Elements in Analysis and Design 40 (7), 793 - 805

Relex [Online],2006, Available: http://www.relexsoftware.com. (date accessed: 4/2007)

ReliaSoft [Online],2006, Available: http://www.reliasoft.com. (date accessed: 4/2007)

REMM [Online],2006, Available: http://www.remm.org. (date accessed: 4/2007)

Rezayat, M., 2000, "The Enterprise-Web Portal for Life-Cycle Support," Computer - Aided Design 32, 85-96.

Rosenman, M. A., Gero, J. S., 1996, "Modeling Multiple Views of Design Objects in a collaborative CAD environment," Computer-Aided Design 28(3), 193-205.

Schenck, S., Wilson, P., 1994, "Information Modeling: the EXPRESS way", Oxford University Press., New York.

Shalev, D. M., Tiran, J., 2007, "Condition-Based Fault Tree Analysis (CBFTA): A New Method for Improved Fault Tree Analysis (FTA), Reliabiity and Safety Calculations," Reliability Engineering and System Safety 92, 1231-1241.

Shellgren, U., Drogou, R., 1998, "Behavior Modeling in Mechanical Engineering - A Modular Approach," Engineering with Computers 14, 185-196.

Snook, I., Marshall, J. M., Newman, R. M., 2003, "Physics of Failure As an Integrated Part of Design for Reliability," *Reliability and Maintainability Symposium*, 46-54.

Solomalala, P., Saiz, J., Mermet-Guyennet, M., et al., 2007, "Virtual Reliability Assessment of Integrated Power Switches Based on Multi-Domain Simulation Approach," Microelectonics Reliability 47(9-11), 1343-1348.

SORMAN [Online],2006, Available: http://www.sorman.com/products/rodon/index.asp. (date accessed: 4/2007)

Spyns, P., Meersman, R., Jarrar, M., 2002, "Data Modeling versus Ontology Engineering," *ACM SIGMOD* 31(4), 12-17.

Steinberg, D. S., 2001, "Preventing Thermal Cycling and Vibration Failures in Electronic Equipment", Wiley, New York.

Sudarsan, R., Fenves, S. J., Sriram, R. D., Wang, F., 2005, "A Product Information Modeling Framework for Product Lifecycle Management," Computer-Aided Design 37, 1399-1411.

SUN [Online],2006, Available: http://java.sun.com. (date accessed: 4/2007)

Suzuki, T. [Online],2005, Available: http://www.onboard-technology.com/pdf_aprile2005/040501.pdf. (date accessed: 4/2007)

SysML [Online],2006, Available: http://www.omgsysml.org/. (date accessed: 4/2007)

Tamburini, D. R.,1999,"The Analyzable Product Model Representation to Support Design-Analysis Integration," *Ph.D. Thesis in Mechanical Engineering. Atlanta: Georgia Institute of Technology*.

Thakkar, D. [Online],2005, Available: http://www.aero.iisc.ernet.in/dipali/synopsis.htm. (date accessed: 4/2007)

Tummala, R. R., 2001, "Fundamentals of Microsystems Packaging", McGraw-Hill, New York.

Tunga, K.,2004,"Experimental and Theoretical Assessment of PBGA Reliability in Conjunction with Field-Use Conditions," *Master Thesis, in Mechanical Engineering. Atlanta: Georgia Institute of Technology*.

Vemuri, K. K., Dugan, J. B., Sullivan, K. J., 1999, "Automatic Synthesis of Fault Trees for Computer-Based Systems," IEEE Transactions on Reliability 48(4), 394-402.

W3C [Online],2003, Available: http://www.w3.org/XML/. (date accessed: 4/2007)

Webster's College Dictionary, 1997, "Random House Webster's College Dictionary", Random House, New York.

Wong, A., Sriram, D., 1993, "Shared Workspaces for Computer-Aided Collaborative Engineering," IESL93-06, Intelligent Engineering Systems Laboratory, Department of Civil Engineering, MIT.

Wood, A. P., 2001, "Reliability-Metric Varieties and Their Relationships," *Reliability and Maintainability Symposium*, 110-115.

Zhao, Z., Shah, J. J., 2005, "Domain Independent Shell for DfM and Its Application to Sheet Metal Forming and Injection Molding," Computer-Aided Design 37, 881-898.

VITA

Injoong Kim was born in Incheon, Korea in 1970. He received his B.S. and M.S. in Mechanical Engineering at Yonsei University, Seoul, Korea in 1996. Throughout his research for the Master's degree, he contributed to the development of the first-ever grade ball screws, which are core components of precision machines and measurement systems.

After finishing the Master's course, he joined the computer-aided engineering team at Samsung Heavy Industries, Daejon, Korea. He developed engineering simulators for design such as performance simulators of cooling fans and dynamic stability simulators of excavators, which gave him the third place in Technical Part Contest at Samsung Heavy Industries in 1997. He also contributed to developing electronic approval systems for ship structures using STEP as part of a joint project with KORDI, KAIST, KR, SAMSUNG and DAEWOO. In 2000 he transferred to Orominfo, Daejon, Korea and developed web-based engineering component management systems.

In September 2001 he joined the Ph.D. program in Mechanical Engineering at the Georgia Institute of Technology, Atlanta, and worked as a graduate research assistant for Dr. Fulton and Dr. Peak. After the untimely passing of Dr. Fulton in 2004, Dr. Sitaraman became his co-advisor. During his PhD studies, I. Kim carried out many industry projects related to information standards and knowledge-based engineering frameworks, and he published conference and journal papers and received two best paper awards. Now he is researching on knowledge-based system design for reliability. His current research interests include product lifecycle management (PLM) for reliability and information-based quality control.